بسم الله الرحمن الرحيم

تعلم كيفية التعامل مع الوينسوك

شرح / الاستاذ . رغيد الطيب

اعداد و تنسيق و تعديل / علي اللوماني

Modi Link Network

www.modilink.com

www.vb4arab.com/vb

ملاحظة : سوف تجد مع هذا الكتاب عدد من الامثلة لتسهيل الدروس عليك

هذه الدروس أخذت من :

http://www.vb4arab.com/vb/showthread.php?threadid=17975

مقدمة

السلام عليكم ورحمة الله وبركاته...

اخوتي الكرام قرأت الكثير من الاستفسارات عن الاداه Winsock وكنت حتى وقت قريب بل منذ اسابيع لا اعرف شيئاً عنها ولكن من الله علي بفهم بسيط عنها ... وسوف احاول هنا نقل كل ما لدي عن هذة الاداه اليك اخي على ان اساهم في رفع غموضها عنك اخي كما ساهم بعض الاخوة في رفع غموضها عني ..... وبسم الله الرحمن الرحيم نبدء:

Winsock هي اداه تستخدم لربط برنامجين مع بعضهما البعض سواء كان هاذان البرنامجان في كمبيوتر واحد ام في كمبيوترين مختلفين وسواءً في شبكة داخلية او خارجية ....

والغرض من هذا الربط هو نقل البيانات من طرف الى طرف آخر وقد يكون الغرض من وراء نقل البيانات هو غرض مفيد او غير مفيد كا التجسس على الآخرين....

ولكي نبدء بالبرمجة مع هذة الاداه يجب علينا اولاً ان نعرف ما هي الشروط الاساسية والاحتياجات لعملية الربط اهم هذة الشروط هو معرفة الاي بي IP وهو عبارة عن رقم يقوم مزود الانترنت لديك بمنحك اياه عند دخولك الشبكة وهو الرقم الذي يمييز جهازك عن باقي الاجهزة داخل الشبكة ولا يمكن ان يتساوى جهازان في الاي بي عند اتصالهما وهو رقم غير ثابت اي انك اذا دخلت الشبكة يتم اعطائك رقم فاذا قمت بالخروج ثم العودة مرة اخرى ستجد ان هذا الرقم قد تغيير ما يهمنا معرفته هو ان هذا الرقم ضروري جداً للربط بين الاجهزة ويمكن الاستعاظة عنه فيي حالة الشبكة الداخلية باسم الجهاز الذي نريد الاتصال به ..... وبشكل افتراضي اذا كنت غير مرتبط في الشبكة الان فان رقم الاي بي التابع لجهازك هو ١٢٧,٠,٠,١ وعند اتباطك فانه يتغيير بحسب ما يعطى لك من قبل مزود الخدمة لديك ....

والشرط الثاني هو المنفذ او Port حيث يمكن للبرنامج استخدام اي منفذ الى اكثر من ستين الف منفذ وهو يستخدم للتراسل بين البرامج مثلاً اذا جعلت برنامج في جهاز A وآخر في جهاز B واردت ان يتصل البرنامج A بالبرنامج B يجب عليك اولاً ان تجعل البرنامج B ينتظر الاتصال على منفذ معين مثلاً المنفذ رقم ١٠٠٠٠ ثم تجعل البرنامج A يطلب الاتصال بالجهاز الذي يمتلك رقم IP التابع للبرنامج B فيقوم البرنامج بطلب الاتصال عن طريق الاداه السابقة الذكر ولكن قبل ذلك يجب ان تحدد المنفذ الذي تريد الاتصال به والذي يجب ان يكون ايضاً ١٠٠٠٠ وعندما يبدء البرنامج A الاتصال على المنفذ المذكور يجد ان البرنامج B منتظراً للاتصال على هذا المنفذ فيحصل بذلك الاتصال وعند هذة النقطة يمكن فعل الكثير....

فمثلاً في برامج التجسس نسمع كثيراً عن مصطلح السيرفر او الباتش وهو في مثالنا البرنامج الثاني حيث يقوم باختيار احد المنافذ ثم الانتظار على هذا المنفذ وعندما تتصل بالشبكة يعمل الهكر على معرفة الاي بي التابع لجهازك ثم يطلب الاتصال به على نفس المنفذ الذي ينتظره فيه برنامج السيرفر وبذلك يحصل الاتصال بينهما ثم يقوم حينها الهكر بسرقة كل البيانات التي يريدها من جهازك ثم ارسالها الى البرنامج الآخر على جهازه .....

هذا هو مفهوم بسيط لفكرة الاتصال وسوف اتطرق لكيفية تطبيق ذلك على شكل دروس بإذن الله تعالى ....

اعداد الأداة

فقد تعبت كثيراً حتى تكونت لدي فكرة بسيطة احاول ان انقلها لكم اخوتي بالتدريج بحيث نفهم الموضوع بشكل عام ونظري اولاً ثم نقوم بالبدء في البرمجة وعندها فقط اثق في انكم جميعاً سوف تنتجون برامج اجمل واقوى مما قمت به انا فلستم بحاجة الا الى وضع ارجلكم على اول الطريق لتنطلقوا تسابقون الرياح وهذا ما اسعى اليه

بعون الله تعالى....
ودعونا الان نبدء بالخطوة التالية في فهم اساسيات تفكيرنا بالبرنامج .... حيث اننا عندما نقوم بكتابة برامج الترابط نقوم بتطوير برنامجان في وقت واحد وليس برنامج واحد احدهما Server او الباتش وهو الذي ينتظر الاتصال والآخر هو العميل او Client وهو الذي يقوم بطلب الاتصال وسوف نسميهما نحن A و B لسهولة استخدمها في الشرح ....
قبل كل شيء يجب ان نضيف الاداه Winsock الى برنامجنا وذلك عن طريق النقر على Components من القائمة Project ثم اختيار العنوان [Microsoft Winsock Control 6.0] من مربع الحوار الذي سيظهر .... وبذلك نكون قد اضفنا الاداة Winsock الى برنامجنا .....
واليكم الان بعض من الخواص المهمة في هذة الاداه والتي يجب علينا فهمها قبل كل شيء :

```
LocatPort               وهو يحدد رقم المنفذ المحلي
LocalIP          يعطي رقم الاي بي المحلي
RemotePort       رقم المنفذ في الجهاز الآخر
RemoteHost       رقم الاي بي للجهاز الآخر
Protocol
نوع الترابط بين البرنامجين ويوجد نوعان نتكلم عنهما لاحقاً
State
يعطي حالة البرنامج الان هل هو متصل ام لا ام منتظر اتصال وهكذا
```

وكذلك توجد اوامر اخرى مهمة جداً مثل :

```
Listen           تستخدم لجعل البرنامج ينتظر الاتصال
GetData          تستخدم لقراءة البيانات التي ارسلت الى البرنامج
SendData                تستخدم لارسال البيانات الى البرنامج الآخر
Connect          تستخدم لطلب الاتصال
Accept           تستخدم لقبول الاتصال
```

وبيقى اخيراً مجموعة من الاحداث المهمة وهي:

```
Connect                  يفيد بان عملية الربط قد تمت
Close                    يفيد بان الربط مفقود
DataArrival
يعلم البرنامج بوصول بيانات من البرنامج الآخر
SendComplete             يفيد البرنامج ان الارسال قد انتهى
Error                    ينبه بوجود خطأ ما
ConnectionRequest
        يفيد بانه يوجد برنامج يريد الاتصال ببرنامجنا
```

طبعاً يبدو الان الامر قد تعقد وانه يجب علينا حفظ الكثر حتى نستخدم الاداه ولكني اريد أن أعلمكم ان الامر ليس كذلك فانت لكي تقوم بعملية ربط برنامجيين ببعض يلزمك فقط سطر واحد من الشفرة في البرنامج الاول وسطران في البرنامج الآخر فقط !!! طبعاً هذا يفيد بعملية الربط ويزيد الكود قليلاً اذا اردت ان ترسل رسالة من البرنامج الاول الى البرنامج الآخر لذى لا اريد ان تصاب اخي بخيبة امل من الكثرة التي تجدها هنا لأني ذكرتها فقط حتى اذا استخدمتها لاحقاً امكنك ان تعود الى هذة البيانات لمعرفة الغرض الاساسي من الامر اي فقط كمرجع لذى من الان وصاعداً هيئ نفسك لنبدء اول عملية ربط بين برنامجينا .... وسوف اجعلها بيسطة جداً لكي يتحمس الذي لم يجرب ذلك من قبل واتمنى ان لا تحبط الذي قام بالربط مسبقاً لاننا سوف نتدرج كي نصل الى برامج كبيرة بإذن الله وتوفيقه ...

الدرس الأول :

بسم الله الرحمن الرحيم ....

سوف نبدء بتصميم اول برنامج وهو عبارة عن برنامج يقوم بارسال رسالة من البرنامج(أ) الى البرنامج (ب) حيث ان البرنامج(أ) هو العميل والبرنامج(ب) هو السيرفر وكنا قد علمنا ان العميل هو الذي يقوم بالاتصال

والسيرفر ينتظر هذا الاتصال ...

تصميم البرنامج(أ) :
==========
نقوم اولاً بفتح مشروع جديد ثم نضع عليه مربع نص نسميه txtMsg ونضيف زرين الاول نجعل عنوانه اتصال والآخر ارسال ثم نضيف اخيراً اداه Winsock لكي تقوم بالاتصال ونجعل اسمها.... wsk

ثم نكتب الشفرة التالية للزر اتصال :

```
Wsk.Close
Wsk.RemoteHost = "127.0.0.1"
Wsk.RemotePort = 10000
Wsk.Connect
```

في الشفرة السابقة قمنا اولاً باستدعاء الامر Close والذي يقوم بقطع الاتصال الحالي حيث انه اذا اردنا ان نبدء اتصال جديد وكان البرنامج اصلاً متصل سوف يسبب ذلك خطأ لذى نقوم اولاً بقطع الاتصال السابق ثم نطلب اتصال جديد ولقد عرفنا سابقاً اننا لكي نتصل بجهاز آخر يجب ان نعرف رقم الاي بي الخاص بذلك الجهاز ونضعه في الخاصية RemoteHost اي انني اريد الاتصال بالكمبيوتر الذي رقمه هو "١٢٧,٠,٠,١" وكنا قد علمنا ايضاً ان هذا الرقم "١٢٧,٠,٠,١" هو الرقم الافتراضي لجهازك اي اننا نخبر الاداه wsk بان تقوم بالاتصال بجهازك نفسه لان هذا هو رقمه بحيث يتسنى لنا التجربة ثم اذا اردنا بعد ذلك ان نتراسل مع جهاز آخر نكتب رقم الاي بي التابع له بدل هذا الرقم ....
ثم قمنا ايضاً بتحديد رقم المنفذ الذي نريد الاتصال به وهو في مثالنا ١٠٠٠٠ ثم اخيراً قمنا بطلب الاتصال بواسطة الامر ..... Connect اذا يمكننا قرأه الاوامر التي ارسلنها الى الاداه wsk كالتالي :

"قم اولاً باغلاق الاتصال السابق ثم قم بالاتصال بالجهاز الذي يمتلك الرقم ١٢٧,٠,٠,١ على المنفذ ١٠٠٠٠"

يمكن ايضاً الاستعاضة عن السابق بالامر التالي اختصاراً :

```
wsk.Close
wsk.Connect "127.0.0.1" , 10000
```

وسيقوم بنفس العمل السابق....

وبهذا نكون قد قمنا بتنفيذ كود الاتصال في برنامج العميل وبقي ان نمكن البرنامج من ارسال رسالة الى البرنامج (ب) بحيث نرسل النص الموجود في مربع النص txtMsg لذا نكتب الاوامر التالية في الزر ارسال :

```
wsk.SendData "" : رسالة من أ & txtMsg.Text
DoEvents
```

علمنا سابقاً انه اذا اردنا ان نرسل شيئاً نستخدم الامر SendData وقد قمنا هنا بارسال العبارة "رسالة من أ : " ثم محتويات مربع النص ... txtMsg اما الامر DoEvents فهو يضع الفرصة للرسالة حتى يتم ارسالها اي يقوم بعملية انتظار انتهاء الارسال ..... وبذلك نكون قد انتهينا من البرنامج الاول ....

- ----------

تصميم السيرفر (ب) :
==========
تختلف طريقة برمجة السيرفر قليلاً وذلك ان المبدء مختلف حيث انه في(أ) وضعنا زر يقوم بطلب الاتصال ثم زر آخر يرسل رسالة اما هنا نحن لن نقوم بطلب الاتصال وانما سوف نقوم بانتظار الاتصال وكذلك انتظار اي رسائل من البرنامج(أ) .. وسوف نجعله بسيطاً جداً بحيث يعرض فقط الرسائل التي تأتيه من البرنامج(أ) ... وسوف نستخدم الاحداث التابعة للاداه Winsock كثيراً هنا لذى يمكنك الرجوع الى قائمة

الاحداث السابقة للتذكر ....

الان ابدء مشروع جديد ثم ضع الاداه Winsock عليه وسميها wsk للاختصار ....

كما نعلم جميعاً فان دور السيرفر الاساسي الان هو انتظار الاتصال على المنفذ الذي سوف يتصل عليه البرنامج(أ) وهو في مثالنا المنفذ رقم ١٠٠٠٠ اذا سوف نختار الحدث Load التابع للفورم لكي نجعله ينتظر الاتصال فور تشغيل البرنامج :

```
Private Sub Form_Load()
        wsk.LocalPort=10000
        wsk.Listen
End Sub
```

في السطر الاول حددنا المنفذ الذي سوف نستمع الى الاتصال منه وذلك بجعل LocalPort=10000 ثم قمنا باستدعاء امر الانتظار او الاستماع .... ونلاحظ هنا انه في البرنامج الاول استخدمنا الخاصية RemotePortلتحديد المنفذ بينما في السيرفر استخدمنا الخاصية LocalPort لذلك وجب الانتباه لهذا.

اذا يمكن قرأة اوامرنا السابقة كالتالي : "قم بانتظار الاتصال على المنفذ المحلي رقم ١٠٠٠٠"

نلاحظ انه عندما تقوم بالضغط على الزر اتصال في البرنامج(أ) انه قبل ان يتم الاتصال يتم اولاً اعلام السيرفر بطلب الاتصال ولن يتم الاتصال حتى يقبل السيرفر هذا الاتصال بواسطة الامر Accept كما اوضحت في جدول الاوامر السابق ....

اذا يتم استخدام الحدث ConnectionRequest التابع للاداه wsk حيث انه هو الذي يفيد بطلب الاتصال من العميل (البرنامج أ) ونكتب داخله اننا نوافق على الاتصال كالتالي :

```
Private Sub wsk_ConnectionRequest(ByVal requestID As Long)
wsk.Close
wsk.Accept requestID
End Sub
```

نقوم اولا باغلاق الاتصال السابق اذا كان موجوداً بالامر Close ثم نقبل الاتصال الجديد بالامر Accept والمتغير RequestID يحتوي على رقم الطلب لذى لا تشغل بالك فيه يكفي ان تكتب ما سبق ....

الان علمنا انه عندما يطلب البرنامج(أ) الاتصال عن طريق الضغط على الزر اتصال اننا سوف نقبل الاتصال وبذلك يتم الاتصال ولكن ماذا يحدث عندما يرسل البرنامج(أ)رسالة لك عن طريق الزر ارسال كما ذكرت سابقاً ؟؟
عندما يستخدم اي برنامج الامر SendData كما فعل البرنامج(أ)لارسال الرسالة فان الحدث DataArival التابع للبرنامج الثاني ينشط للاعلام بوصول بيانات جديدة يمكن اخذها عن طريق GetData لذا بمجرد وصول الرسالة من(أ)ينشط الحدث DataArival في(ب)ويمكنك استخدامه للحصول على البيانات او الرسائل كالتالي :

```
Private Sub wsk_DataArrival(ByVal bytesTotal As Long)
Dim S As String
wsk.GetData S
MsgBox S
End Sub
```

في السطر الاول عرفنا متغير جديد من النوع String باسم S لكي نضع الرسالة فيه ثم استخدمنا الامر GetDataلمعرفة الرسالة ووضعها في المتغير S ولاننا لا نريد ان نعمل بهذة الرسالة اي شيء سوى ان نعرضها قمنا باستخدام الامر MsgBox لعرض مربع حوار به هذة الرسالة .....

هنا فقط نكون قد انتهينا من البرنامج(ب)يمكن الان ان نشغل البرنامج(أ)ثم نشغل البرنامج(ب) بحيث يعملان سوياً وبعد ذلك ننقر على الزر اتصال في(أ)حتى يحدث الاتصال ثم اخيراً نكتب الرسالة التي نريدها في مربع النص في(أ)ونضغط على الزر ارسال فنرى الرسالة التي كتبناها قد ظهرت في البرنامج(ب).... وهذا ما يحدث بين جهاز وآخر بحيث نرسل رسالة من كمبيوترك الى كمبيوتر آخر بعيد بنفس الطريقة المهم هو رقم الاي بي وكذلك ان يكون برنامج السيرفر شغال عنده بحيث اذا طلب البرنامج(أ)الاتصال يجد البرنامج(ب) ينتظر اتصاله فيقبله ويحدث بذلك الاتصال....

طبعاً المثال بسيط ولم نراعي فيه تجنب الاخطاء وذلك لتوضيح الصورة مبدئياً لذى في حالة ظهور اي خطأ قم باغلاق البرنامجان واعد تشغيلهما مرة اخرى وابدء الاتصال كما سبق بالنقر على اتصال ثم كتابة رسالة ثم الزر ارسال ولا تضغط على ارسال قبل اتصال اي ان الزر اتصال اضغطه مرة واحدة حتى يتم الاتصال بشرط ان يكون السيرفر شغال حينها حتى يقبل الاتصال ثم ارسل ما بدى لك من الرسائل الى (ب) باستخدام الزر ارسال....

ان اهم شيء فيما سبق ليس البرنامجان وانما فقط مايحدث بالضبط منذ بداية الاتصال وحتى قبول الاتصال وارسال الرسالة و عرضها و و و ....

لذى سوف اقوم بالتذكير بما يحدث بشكل سريع اولا نحدد رقم الاي بي والمنفذ للجهاز الذي نريد الاتصال به عن طريق RemoteHost و RemotePort ثم نقوم بطلب الاتصال بالامر Connect ولان البرنامج الذي نريد الاتصال به (البرنامج ب) موجد على نفس الجهاز استخدمنا رقم الاي بي الافتراضي ١٢٧,٠,٠,١ ..... ولكي نرسل رسالة استخدمنا الامر ... SendData هذا في كله البرنامج(أ)...

اما في البرنامج(ب) فقد قمنا اولاً بانتظار الاتصال وحددنا المنفذ عن طريق LocalPort ثم امر الاستماع او الانتظار Listen وبذلك نكون قد انتظرنا الاتصال ولكن عندما يطلب الطرف الاخر الاتصال ينشط الحدث ConnectionRequestلاعلامنا بذلك فنقوم بقبول او رفض هذا الاتصال ولاننا نريد الاتصال فقد قبلناه باستخدام الامر Accept وبذا يكون الاتصال قد تم ولكن بقي انه عندما يرسل الطرف الاخر بيانات او رسائل ينشط الحدث DataArival لاعلامنا بذلك فنفعل بالبيانات ما نريد بعد اخذها بالامر GetData وهذا كل شيء حتى الان....

اتمنى ان تكون مكنيكية العمل قد فهمت بشكلها الصحيح لانه من هذة الاساسيات يمكن الانطلاق الى مدى بعيد ...

الدرس الثاني

والان دعونا اخوتي نناقش خاصية مهمة جداً في هذة الاداه وهي الخاصية State والتي نستطيع بواسطتها معرفة حالة الاتصال بين البرنامجين مثلاً اذا كنت قد عملت اتصال بين(أ) و(ب)وكان البرنامجان في اجهزة منفصلة ثم قمت بتبادل الرسائل وقام الطرف الاخر باغلاق برنامج المحادثة اثناء التشغيل في مثالنا السابق اذا قمت انت بارسال رسالة وكان البرنامج الآخر قد اغلق لاي سبب كان مثل (اعادة تشغيل الجهاز او عن غير قصد المهم انه مغلق) فانه سوف تظهر عندك رسالة خطأ ثم يتوقف برنامجك بينما في الماسنجر مثلاً فأنه يتم ابلاغك بان الرسالة لن تصل الى المستخدم الآخر لذى سوف نسعى الان لوضع معلومة بيسطة عن حالة الاتصال بحيث تستطيع من خلالها ان تميز اذا كان بامكانك ان ترسل رسالة ام لا ....

طبعاً الذي يفيدنا هنا هو الخاصية State حيث ان لها قيم مفيدة لتعطي حالة الاتصال الحالي وهي كالتالي :

```
State = 0               مغلق
State = 1               مفتوح
State = 2               جاري انتظار الاتصال
State = 3               الاتصال معلق الان
State = 4               الطرف الآخر يعالج البيانات
State = 5               الطرف الآخر انهى المعالجة
State = 6               جاري الاتصال الان
State = 7               متصل
State = 8               قام الطرف الآخر بقطع الاتصال
State = 9               حصل خطأ ما
```

يتضح مما سبق انه اذا فحصنا الخاصية State وكان الناتج = ٧ فان ذلك يعني ان البرنامجان متصلان اما غير ذلك فانهما غير متصلان او حالة انتظار ....

ويمكن استغلال هذة الخاصية الجميلة في معرفة ما اذا كان المتصل قد قام باغلاق الاتصال .... كما انه كتطبيق جيد ايضاً ان نجعل Label يظهر للمستخدم حالة الاتصال الحالية ...

كما ان هناك امر مهم اذا قام العميل باتصال مع السيرفر وبعد ان نجح الاتصال قام شخص باغلاق برنامج العميل فجئة ثم اعاد تشغيله بعد ذلك وضغط على الزر اتصال هل سيحدث الاتصال طبعاً لا .... لماذا ؟ وذلك لانه عندما تم تشغيل السيرفر كانت حالته ٢ (يعني منتظر اتصال انظر الى الجدول اعلاه) وذلك لاننا استخدما الامر Listen في بداية التشغيل وعندما قام العميل بطلب الاتصال كان السرفر بانتظاره فحصل الاتصال بينهما وبالتالي فان حالة البرنامجان عندها اصبحت ٧ (اي متصل) اما عندما قام العميل بلاغلاق فجئة اصحبت حالة السيرفر ٨ (تنبية باغلاق الطرف الآخر) اي انه ليس في حالة انتظار فلما حاول العميل بعد ذلك الاتصال مرة اخرى لم يكن السيرفر بانتظاره ففشلت عملية الاتصال .... جرب البرنامج في الدرس السابق وسوف ترى ذلك الخطأ .....

طبعاً الحل بسيط وذلك كله باستخدام هذة الخاصية الرائعة State حيث يمكننا مثلاً ان نضع مؤقت Timer يقوم بعرض حالة الاتصال للمستخدم بحيث اذا صادف ان الحالة اصبحت ٨ (اي قطع الاتصال) او ٩ (اي حدث خطأ ما) يقوم السيرفر بالتالي:

- 1اظهار رسالة للمستخدم بانتهاء الاتصال ..
- 2اغلاق الاتصال السابق الذي حصلت فيه مشلكة باستخدام الامر.... Close
- 3يقوم بتحديد المنفذ من جديدعن طيرق... LocalPort
- 4يستدعي الامر Listen بحيث يعود الى حالة انتظاره مرة اخرى كي ينتظر اتصال العميل ..

```
Select Case wsk.State
 Case 0: Label1.Caption = ""الاتصال مغلق
 Case 1: Label1.Caption = ""الأتصال مفتوح
 Case 2: Label1.Caption = ""جاري انتظار الاتصال
 Case 3: Label1.Caption = ""الاتصال معلق الان
 Case 4: Label1.Caption = ""الطرف الآخر يعالج البيانات
 Case 5: Label1.Caption = ""الطرف الآخر انهى معالجة البيانات
 Case 6: Label1.Caption = ""جاري الاتصال الان
 Case 7: Label1.Caption = ""متصل
 Case 8: Label1.Caption = ""يقوم الطرف الآخر بقطع الاتصال
 Case 9: Label1.Caption = ""خطأ
End Select

' نتاكد من وجود خطا فنقوم باعادة الانتظار
If wsk.State = 8 Or wsk.State = 9 Then
   MsgBox ""قام الطرف الاخر بقطع الاتصال او انه يوجد خطأ في الاتصال
   wsk.Close
   wsk.LocalPort = 10000
   wsk.Listen
End If
```

لاحظ اننا استخدمنا العبارة Select Case وذلك لاظهار حالة الاتصال للمستخدم بحيث يكون على دراية تامة بما يحصل الان في عملية الاتصال ....

طبعاً كل ما سبق يعطي برنامجنا شيئاً من القوة ولكن بقي امر لم نقوم به فماذا اذا قام احد البرنامجين بالضغط على الزر ارسال قبل ان يكون الاتصال قد تم فعلاً اي قبل الضغط على الزر اتصال في العميل النتيجة طبعاً هي خطا فيكف يمكن ان نستخدم SendData لارسال البيانات قبل ان يكون هناك اتصال فعلاً .... وما العمل في هذة الحالة ؟؟؟؟؟

طبعاً العمل بسيط وهو التاكد بان هناك اتصال قبل ارسال البيانات عن طريق ... State حيث كانت الشفرة في الزر ارسال كما يلي:

```
Wsk.SendData "": أ من رسالة & txtMsg
DoEvents
```

ويجب ان تصبح هكذا :

```
    If wsk.State = 7 Then
        wsk.SendData "" : أ من & txtMsg
        DoEvents
    End If
```

وذلك في كلاً من العميل والسيرفر بحيث يتم التحقق من هناك اتصال فعلاً ثم ارسال البيانات بعد ذلك .

بقي نقطة اضافية في الدرس السابق حيث كنا عندما يقوم المرء باضغط على الزر اتصال نقوم باغلاق الاتصال السابق ثم نطلب اتصال جديد وذلك كالتالي :

```
    wsk.Close
    wsk.RemoteHost = "127.0.0.1"
    wsk.RemotePort = 10000
    wsk.Connect
```

كان ذلك قبل ان نتعرف على الخاصية State اما الان فما هو الداعي لاغلاق الاتصال بالامر Close اذا كان الاتصال يعمل بشكل سليم !!!! لذى سوف نتأكد اولاً اذا كانت الحالة هي ٧ اي الاتصال قائم اذا لا داعي لفعل شيء اما غير ذلك فانه يوجد خطاء او ماشابه كأن يقوم الخادم بالاغلاق فيجب عندها ان نغلق الاتصال الحالي ثم نطلب اتصال جديد ... كالتالي :

code:

```
    If wsk.State <> 7 Then
       wsk.Close
       wsk.RemoteHost = "127.0.0.1"
       wsk.RemotePort = 10000
       wsk.Connect
    End If
```

(العلامة <> تعني لايساوي)
وبذلك نجدد الاتصال اذا كان الاتصال الحالي به قصور فقط ...

اخوتي ان المافهيم السابقة على الرغم من بساطتها فهي مهمة جداً في التخلص من رسائل الخطأ التي قد تنتج دائماً من استخدام Winsock حيث يجب ان نكون متأكدين من وجود اتصال قبل اي عملية لتبادل البيانات كما انه يجب ضمان ان يعود الخادم (السيرفر) الى حالة انتظار اذا ما حصل اي خطأ في الاتصال وهو ما قمنا به سوياً في هذا الدرس لذى يجب على الجميع اتقانه جيداً قبل الاستمرار في الدروس القادمة...

الدرس الثالث

سوف نبني في هذا الدرس اول برنامج للمحادثة بين شخصين يشبه الماسنجر في العمل العام ... حيث تظهر الرسائل التي تتلقيها من الطرف الآخر على شكل قائمة بها اسمه ثم الكلام الذي بعثه ....

برنامج العميل :

نقوم باضافة ثلاثة مربعات نص للبرنامج السابق ونسميهم :

```
txtName
وهو المربع الذي سوف يحمل اسمك في الحوار
```

```
txtIP
المربع الذي سيحتوي على رقم الاي بي الذي نريد الاتصال به
txtList
مربع يحتوي على الرسائل المرسلة والمستقبلة
```

ونجعل الخاصية MultiLine=True لمربع النص txtList بحيث يسمح بادخال اكثر من سطر بحيث تبدو على شكل قائمة يوضع فيها الحوار الذي يدور بين العميل والخادم ...
والذي سوف يتغير هو انه عندما تصلنا رسالة من الخادم فاننا نريد اضافتها الى اسفل القائمة وليس عرضها بمربع حوار لذى نستبدل الكود في الحدث DataArival كالتالي :

```
    Dim S As String

    wsk.GetData S

    txtList = txtList & vbNewLine + vbNewLine + S

    txtList.SelStart = Len(txtList)
```

بعد قراة البيانات المرسلة وجعلها داخل المتغير S نقوم بوضعها في اسفل القائمة وذلك بعد سطرين فارغين vbNewLine والذي يمثل سطر جديد وفي الاخير نريد ان نسحب التركيز الى اسفل القائمة وذلك بالسطر txtList.SelStart = Len(txtList)i وبدونه سوف تمتلىء القائمة ثم يغيب النص الجديد في الاسفل دون ان نراه كما هو ملاحظ في الماسنجر عندما يقوم الشخص المتحدث بارسال رسائل لك فان شريط التمرير ينزل للاسفل بحيث يعرض لك آخر ما تم ارسالة من نص وهذا ما يحدث بالضبط بواسطة هذا السطر ....

كما انه عندما يضغط احد على الزر ارسال يتم ارسال النص المكتوب في المربع txtMsg ويكون الاسم المكتوب في المربع txtName امامه بحيث يرى الاسم الذي وضعته لنفسك امام النص وذلك كله في الزر ارسال كالتالي :

```
If wsk.State = 7 Then
wsk.SendData txtName + " : " + txtMsg
DoEvents
txtList = txtList & vbNewLine & vbNewLine & txtName & " : " & txtMsg
txtMsg = ""
End If
```

بقي شيئاً واحداً هو انه عند الضغط على Enter يجب ان يتم ارسال النص وكانك نقرت على الزر ارسال وذلك يكون بفحص الزر المضغوط عليه في الحدث KeyPress التابع لمربع النص فاذا كان الزر هو Enter يتم النقر على الزر ارسال لارسال البيانات .. وذلك كما يلي :

```
Private Sub txtMsg_KeyPress(KeyAscii As Integer)
If KeyAscii = 13 Then Command1 = True
End Sub
```

حيث ان الرقم ١٣ هو رقم الزر Enter وبذلك نكون قد انهينا برنامج العميل ....

حيث ان كل زر على لوحة المفاتيح له رقم خاص و لكنك غير ملزم بحفظ ارقام جميع المفاتيح حيث يوفر لك الفيجوال بيسك اختصارات لتلك المفاتيح مثلا بدل من ان نكتب الرقم ١٣ نكتب vbKeyReturn و اذا اردنا المفتاح D مثلا نكتب vbKeyD و هكذا ...

اما الخادم فهو بنفس التعديل الذي قمنا به في العميل مع فرق بسيط حيث انه لاداعي لاضافة مربع لوضع رقم الاي بي لانه لن يتم الاتصال من الخادم فقط يقوم بالانتظار ....

الدرس الرابع

اما الان دعونا نفكر بشي من التفصيل نحن دائماً نقوم بارسال رسائل نصية ولكن كيف يمكنني ان اؤدي مهام غير عرض الرسائل او لنقل مبدئياً اذا كان لدينا برنامج للعميل وفيه مربع نص txtMsg واربعة ازرار واحد هو للاتصال ... اما الثلاثة الآخرون فاسمائهم :

1 - Text
2 - Message
3 - Caption

واريد انه اذا نقر على الزر الذي اسمه Text يتم عرض الرسالة التي في مربع النص txtMsg في مربع نص اسمه Text1 داخل الخادم ....

واذا نقر المستخدم على الزر Message فان الرسالة تعرض بواسطة مربع حوار في برنامج الخادم ....

وكذلك عند النقر على الزر Caption في العميل يتم عرض الرسالة في شريط عنوان الخادم اي عنوان الفورم التابعة لبرنامج الخادم ...

طبعاً هنا يختلف الوضع فنحن لانريد من العميل ان يخبرنا ماهي الرسالة فقط كما كان سابقاً حيث نضعها في المتغير S ثم نعرضها ولكننا الان نريد من العميل ان يخبرنا كيف يريد ان يتم عرض هذة الرسالة ايضاً .... وبذلك نحن نحتاج الى نوعين من المعلومات وهي نوع العرض و نص الرسالة ....

اي ان الطريقة السابقة لاتنفع وهي مثلاً :

```
Wsk.SendData txtMsg
DoEvents
```

فهذة الطريقة ترسل النص الموجود في مربع النص txtMsg ولكنها لا تخبر الخادم كيف يعرض هذا النص

وهنا نبدء في تغيير طريقة الارسال وطريقة الاستلام للبيانات في كلاً من الخادم والعميل..

والفكرة هي ان نضع عدد معين من الحروف ولنقل اربعة قبل كل رسالة تبين نوع العرض مثلاً :

```
Text    تعني قم بوضعها في مربع نص
Cptn    تعني قم بوضعها في عنوان الفورم
MgBx    تعني قم بوضعها في مربع حوار
```

مثلاً سيكون شكل الكود في الزر Message في برنامج العميل بالشكل التالي :

```
Wsk.SendData "MgBx" + txtMsg
```

وبذلك عندما يقوم الخادم بإستلام الرسالة يفحص الحروف الاربع الاولى ويقوم بعرض باقي الرسالة بالطريقة التي تخبره به الحروف الاربعة الاولى ...

ويكون ذلك بفصل الرسالة الى جزئين الاول وهو الامر ووضعته في متغير اسمه Cmd والباقي هو نص الرسالة ووضعته في متغير اسمه Msg ثم نقوم بفحص الامر Cmd ونحدد طريقة العرض باستخدام Select Caseكالتالي :

```
Dim S As String
Dim Msg As String
Dim Cmd As String

wsk.GetData S

Cmd = Left(S, 4)
Msg = Mid(S, 5)

Select Case Cmd
 Case "Text":   Text1.Text = Msg
 Case "MgBx":   MsgBox Msg
 Case "Cptn":   Caption = Msg
End Select
```

الامر Left يقوم باقتطاع عدد من الاحرف حسب الرقم الذي وضعناه من يسار المتغير الذي حددنا
مثلا في مثالنا هذا يقوم الامر left بأخذ أول ٤ احرف من المتغير Sو تخزينها في المتغير Cmd
و الأمر Mid يقوم باخذ عدد من الحروف بمقدار بداية و نهاية نحدده نحن و اذا لم نحدد مقدار النهاية يقوم باخذ الحروف من مقدار البداية الى نهاية المتغير
مثلا في مثالنا قمنا باخذ الاحرف من الحرف رقم ٥ الى نهاية المتغير S و تخزينه في المتغير Msg

الدرس على الرغم من سهولته فهو بنظري من اهم المواضيع لفهم تنظيم تبادل البيانات والاوامر بين العميل الى الخادم...

حيث اننا فيما سبق كنا نستخدم مصطلح الخادم مجازياً ولقد سمي الخادم لانه يجب ان يوفر للعميل الخدمات التي يطلبها مثلاً اذا اراد العميل يعرف كم الوقت في الجهاز الذي به برنامج الخادم يقوم بارسال رسالة تبدء بحروف متفق عليها بينهما مثلاً Time وعندما تصل الرسالة الى الخادم يقوم بتفحص الحروف الاولى ويعرف ماذا اراد العميل بالضبط ثم يقوم بمعرفة الوقت بواسطة اوامر الوقت ثم يستخدم الامر SendDataلارسال البيانات التي طلبها العميل منه ....

على العموم سوف نتطرق الى هكذا امور ان شاء الله مستقبلاً .... ونكتفي الان بمثال على عرض الرسالة باحدى الطرق الثلاث السابقة ....

أساسيات من اجل الدرس الخامس

رأيت انني سوف احتاج الى بعضاً من الاساسيات في المرحلة القادمة واخشى ان لا اجدها عند البعض لذى سوف اوضح هنا امرين سوف احتاجهما في برنامج المحادثة الجماعية القادم باذن الله تعالى .. وارجوا ممن ليس لديه فكرة كبيرة فيهما ان يقرأ الشرح ثم يطبق كل شيء بحيث لا تتصعب عليه الامور بعد ذلك وهي امور بسيطة جداً فقط تحتاج الى تطبيق ...

اما الموضوع الاول فهو مصفوفة الادوات واستخدامها او Control Array و الثاني هو صندوق القائمة او الـ ListBoxوطريقة التعامل معها فهي مهمة جداً في الموضوع القادم ولا اريد ان اخلط بين شرح فكرة البرنامج وشرح التعامل مع هاتان الخاصيتان لذى سوف اقدم شرح سريع عنهما اتمنى ان يؤدي الغرض ....

اولاً : مصفوفة الادوات والتي تعني وجود ادوات فوق الفورم من نفس النوع وبنفس الاسم يفرق بينهم بالرقم مثلاً ان يكون عندنا ثلاثة ازرار كلها بإسم Command1 ولكي نفرق بينها نفرق بينها بالرقم او الـ Index مثلا :

```
Command1(0).BackColor = vbRed
Command1(1).Caption = "Welcome"
Command1(2).Width = 800
```

في الكود السابق الزر الاول رقمة ٠ وقمنا بتغيير لونه الى الاحمر والزر الثاني رقمه ١ وقمنا بجعل عنوانه هو Welcome والزر الاخير او الثالث رقمه ٢ وجعلنا عرضة ٨٠٠ ....

واذا نظرنا الى الحدث Click لاي زر فانه قبل ان يكون هناك ازرار بنفس الاسم فهو على الشكل التالي:

```
Private Sub Command1_Click()

End Sub
```

وبعد ان نضيف الى الفورم عناصر اخرى بنفس الاسم فانه يصبح :

```
Private Sub Command1_Click(Index As Integer)

End Sub
```

والملاحظ هنا انه تم اضافة متغير جديد اسمة Index وهو الذي يحدد الزر الذي تم النقر عليه ...
وبذلك فاذا اضفنا مثلاً ثلاثة ازرار بنفس الاسم مع تغيير الخاصية Index اي ان لكل منهم رقماً فانه لن يكون لكل زر من الازرار حدث Click خاص به ولكن كل الاحداث تكون مشتركة ويمكنك تمييز الزر الذي ثم النقر عليه بفحص القيمة Index لنفترض اننا نريد ان نجعل عنوان الفورم One اذا ثم النقر على الزر رقم واحد و Two اذا نقر على الزر رقم اثنان و Three في حالة النقر على الزر الثالث فاننا سوف نكتب التالي:

```
Private Sub Command1_Click(Index As Integer)

    Select Case Index
      Case 0: Caption = "One"
      Case 1: Caption = "Two"
      Case 2: Caption = "Three"
    End Select

End Sub
```

وذلك لان الترقيم يبدء من صفر....
وكل ماسبق كان لاننا سوف نحتاج الى اضافة اكثر من اداه Winsock باسم واحد ولنقل مثلاً wsk بالتالي فاننا سوف نميز بينها بالرقم كما فعلنا مع الازرار السابقة الذكر .....

ثانياً : مربع القائمة وهو ListBox ويستخدم لعرض ببيانات او عناصر في قائمة بحيث يمكنك تحديدها واختيار احداها وعندما تريد ان تضيف عنصر الى القائمة ولنفرض ان اسم القائمة هو List1 فاننا سوف نستخدم الامر AddItem للاضافة لها بالشكل التالي:

```
List1.AddNew "" كلنا عبيد لله سبحانه
```

و الكود السابق سوف يضيف العبارة "كلنا عبيد لله سبحانه" الى مربع القائمة واذا اردنا اضافة المزيد نستخدم نفس الكود...

وكذلك الامر RemoveItem يستخدم لحذف عنصر من القائمة ويجب ان تمرر له رقم العنصر الذي تريد حذفه هل الاول او الثاني او او .... بحيث يبدا الترقيم من الصفر اي انك لكي تحذف العنصر الاول في القائمة اكتب التالي :

```
List1.RemoveItem 0
```

وصفر يعني العنصر الاول بينما ١ يعني العنصر الثاني وهكذا ...

واما الخصائص فهي كما يلي :

```
Text
هي عبار عن نص يحتوي على العنصر المحدد من القائمة
List            عبارة عن مصفوفة بالعناصر كاملة
ListCount               هو عدد العناصر في القائمة
ListIndex
يحمل رقم العنصر المحدد ويكون (-١) اذا لم يكن هناك عنصر محدد
```

وبالتالي لنفترض اننا نريد ان نطبع كل عناصر القائمة في الفورم فاننا سوف نستخدم التالي :

```
    Dim I As Integer

    For I = 0 To List1.ListCount - 1
        Print List1.List(I)
    Next
```

حيث اننا عملنا دوران يبدء من الرقم صفر والذي يعني اول عنصر في المصفوفو الى النصر الذي رقمه هو List1.ListCount-1وهو العنصر الاخير في القائمة لان الترقيم يبدء من الصفر وبالتالي اذا كان هناك ثلاثة عناصر في القائمة اي ان ListCount=3 فان رقم العنصر الاخير هو ٢ وهو.... (ListCount-1)

يجب على الجميع فهم ما سبق جيداً كي اتمكن من ان اشرح الموضوع التالي بدون الالتفات الى هذة المبادىء الاساسية

الدرس الخامس

اخوتي سنبدأ هنا بتوضيح الفكرة العامة للمحادثة الجماعية بشكل سريع ثم نلحقه بعد ذلك بمثال وشرح لهذا المثال حتى نستفيد جميعاً من الناحية النظرية والتطبيقية...

في السابق كنا نستخدم اسم الخادم مجازياً فلم يكون يقوم بدور آخر سوى الانتظار وعندما يتم الاتصال يصبح الخادم له نفس دور العميل في الارسال و الاستقبال وفي الاصل الخادم لم يسمى بهذا الاسم الا لانه يقوم بتقديم خدمات للعميل اي انه يوفر له المعلومات و البيانات التي يريدها العميل كما ينظم سير عملية التواصل بين الاطراف اما في برامج المحادثة السابقة لم نرى الدور الفعلي له ....

اما في موضوع المحادثة الجماعية فان كل مستخدم مشترك في المحادثة سيكون لديه عميل مستقل ولن يكون شرطاً على كل عميل بالاتصال بكل عميل آخر حتى يستطيع تبادل الرسائل بينهما وانما سوف يتصل الجميع بالخادم والذي سكون بدورة متصلاً بالجميع وعندما يقوم احد الاشخاص بارسال رسالة من برنامج العميل الذي بحوزته الى الخادم فان الخادم سيقوم باستلام الرسالة ثم يقوم بدوره بارسالها الى جميع الاعضاء المشتركين في المحادثة وبذلك فان عملية تنظيم المحادثة بين الاطراف سوف تقع على عاتق الخادم الذي سيضمن تواصل الجميع بعضهم ببعض ... ولكن كيف يمكن للخادم القيام بهذا الدور وكذلك ان يكون متصلاً باكثر من جهة اتصال (باكثر من عميل واحد) .... وهذا ما سوف نناقشه معاً الان

سوف تختلف الفكرة الان من حيث ان الخادم لن يرتبط بعميل واحد بل يجب ان يمتلك القدرة على الارتباط با كثر من عميل بحيث يبقى منتظراً للاتصال من اكثر من جهة ولانه ليس للاداه Winsock القدرة على الارتباط باكثر من جهه في نفس الوقت لذى تظهر الحاجة الى ان يمتلك الخادم اكثر من اداه winsock بحيث تقوم بتحقيق الاتصال بكل عميل يحاول الاتصال به (وهنا سوف نحتاج الى مصفوفة من ادوات.winsock

قد يتبادر الى الذهن مما سبق اننا سوف نجعل اكثر من اداه في الخادم بحيث تنتظروا هذة الادوات الاتصال على نفس المنفذ طبعاً لا لانه لايمكن ان تستخدم اكثر من جهة المنفذ ذاته في نفس الوقت فعندما نستدعي الامر Listenتقوم الاداه بحجز المنفذ المحدد بالخاصية LocalPort ولايمكن ان يتم استغلاله بواسطة اداه اخرى الا في حالة ان الاداه غيرت حالتها(State=2 ويعني انتظار) الى اي شيء آخر سوى ان تصبح متصلة او تنهي الانتظار بواسطة Close وعندما يتحقق الاتصال فانه عندما نقبله بواسطة Accept ينتهي الانتظار ويصبح من الممكن استغلال المنفذ مرة اخرى .... وليس من المعقول ان نجعل كل اداه تحجز منفذ مختلف فهذة طريقة مكلفة ومربكة حيث انك سوف تحجز عدد لابأس به من منافذ النظام كما ان العميل عندما سيحاول الاتصال يجب عليه تجربة الاتصال بكل المنافذ المفتوحة فلربما انه طلب الاتصال بالمنفذ ١٠٠٠٠ مثلاً ووجده مرتبط مع شخص آخر يجب عليه بعدها ان يجرب المنفذ ١٠٠٠١ حتى يصل الى منفذ فارغ ويتصل به وهذا حل غير عملي ....

بل ان كل ما سبق كان مجرد حلول ضعيفة ذكرتها حتى نتعايش معاً مع المشكلة ونتعود دائماً على تقليب كل الحلول في رؤوسنا للوصول الى الحل الامثل ....

اما الحل المناسب في رأيي هو ان نقوم بوضع اكثر من اداه winsock في الخادم ولنقل ١١ اداه بحيث نجعل الاداه الاولى اي رقم 0 تقوم بانتظار الاتصالات الخارجية على منفذ محدد يعرفه كل العملاء ويقومون بالاتصال به وقد كنا في السابق نقوم بقبول الاتصال بمجرد وصول طلب الاتصال الينا من العميل كالتالي :

```
Private Sub wsk_ConnectionRequest(ByVal requestID As Long)

    wsk.Close
    wsk.Accept requestID

End Sub
```

اما الان اذا فعلنا ذلك فان الاداه سوف تنهي حالة الانتظار ولن يكون بمقدورها استلام اي اتصال من اي عميل آخر لذى فنحن هنا لن نعمل ذلك .... ولكن الاداه المؤلة عن الانتظار في حالتنا الجديدة هذة لن تقوم بقبول الاتصال مباشرة بل سوف تبحث في الادوات العشر المتبقية بحيث اذا صادفت اداه فارغة اي

غير متصلة انها مغلقة وحالتها State=0 وتجعلها تقبل الاتصال الجديد وبالتالي تبقى هي (الاداه رقم ٠) على حالتها السابقة في انتظار اي اتصال جديد وذلك على النحو التالي :

```
Private Sub wsk_ConnectionRequest(Index As Integer,
ByVal requestID As Long)
    Dim I As Integer

    For I = 1 To 10
        If wsk(I).State = 0 Then
           wsk(I).Accept requestID
           Exit For
        End If
    Next

End Sub
```

لا حظ اننا بدأنا البحث من الاداه رقم واحد متجنبين بذلك الاداه رقم ٠ كي تبقى هي حلقة الوصل بين برامج العملاء والخادم...

اما البيانات التي سوف يتلقاها الخادم من العميل هي عبارة عن احدى رسالتين الاولى تبدء بـ NewMssg كا امر متبوع بنص الرسالة وهو عبارة عن رسالة يبعثها العميل ولان كل عميل هو متصل بالخادم وليس بعميل آخر فانه لن يبعث الرسائل الا الى الخادم وتقع على الخادم مهمة ارسالها الى باقي العملاء بحيث تظهر عندهم بانها من العميل فلان ... حيث يقوم الخادم عند استلامه هذا الامر بالبحث عن جميع العملاء المتصلين اي التي تكون حالة الاداه المرتبطة بهم هي State=7 ويقوم بارسالها اليهم بحيث يقوموا بعرضها في مربعات الرسائل لديهم ...

هذا كان بالنسبة للرسالة الاولى اما الرسالة الاخرى التي يمكن ان يتلقيها الخادم من العميل هي الرسالة NewUser ثم يليها اسم المستخدم فعندما يتم الاتصال بين العميل والخادم يجب على العميل ارسال هذة الرسالة قبل اي شيئ آخر بحيث يليها اسم العميل حتى يقوم الخادم بارسال رسالة الى كل العملاء الآخرون بان العميل المسمى فلان قد اضيف الى المحادثة ..... وذلك في برنامج الخادم كالتالي :

```
Private Sub wsk_DataArrival(Index As Integer, ByVal
bytesTotal As Long)
    Dim Cmd As String
    Dim Msg As String
    Dim S As String
    Dim I As Integer


    wsk(Index).GetData S

    Cmd = Left(S, 7)
    Msg = Mid(S, 8)

  ' ------------------------------------------
    If Cmd = "NewMssg" Then

       For I = 1 To 10
           If wsk(I).State = 7 Then
              wsk(I).SendData "Message" + Msg
              DoEvents
           End If
       Next

    End If

  ' ------------------------------------------
    If Cmd = "NewUser" Then
```

```
        Num = Num + 1

        For I = 1 To 10
            If wsk(I).State = 7 Then
               wsk(I).SendData "MssgBox" + Msg
               DoEvents
            End If
        Next

    End If

End Sub
```

لاحظ ان طول الامر Cmd هنا هو ٧ حروف ... كما يجدر الاشارة بان الخادم عند استلام هذة الرسائل من العميل يقوم بارسال احدى رسالتين الى كل العملاء الاخرون وهذة الرسالتان هما اما MssgBox او Messageيستلم الخادم رسالة NewMsg من احد العملاء يرسل الرسالة Message الى كل العملاء الاخرون الذي يقومو بعرض محتويات هذة الرسالة في قائمة المحادثة ... اما اذا استلم الخادم الرسالة NewUserوذلك عند تسجيل عميل جديد فانه يرسل الرسالة MssgBox الى باقي العملاء كي يقومو بعرض مربع حوار يفيد بان مستخدم جديد قد اضيف الى المحادثة....

وذلك كالتالي في العميل كالتالي :

```
Private Sub wsk_DataArrival(ByVal BytesTotal As Long)
    Dim S As String
    Dim Cmd As String
    Dim Msg As String

    wsk.GetData S
    Cmd = Left(S, 7)
    Msg = Mid(S, 8)


    Select Case Cmd
     Case "Message": txtList = txtList & vbNewLine +
vbNewLine + Msg
     Case "MssgBox": MsgBox "لقد تمت اضافة المستخدم " +
Msg + " الى المحادثة"
    End Select

    txtList.SelStart = Len(txtList)
End Sub
```

نلاحظ انه يوجد فقط نوعين من الرسائل التي يتم تبادلها بين العميل والخادم وهذة طبعاً لا تكفي للقيام بمعظم محاور برامج المحادثة ولنأخذ مثلاً ان المستخدمين لايعلمون كم عدد الافراد الذين يتحدثون معهم الان كما انه اذا غادر احدهم المحادثة فلايعلم الباقون بمغادرته وذلك اننا لم نضمن ذلك في هذا الربنامج الذي تعمدت جعله بسيطاً حتى اتأكد من تسرب الفكرة الى رؤوسنا ثم نناقش بعد ذلك اضافات على الموضوع لذى ارجو ان يقوم الجميع بفهم الدرس جيداً لان الدرس الذي يليه سكون معقداً بشكل اكبر فمن اراد الاستمرار بشكل سليم فعليه فهم كل سطر في البرنامج والمهم هو فهم الفكرة العامة بحيث يمكن تطويرها بعد ذلك لتناسب احياجاته مع اضافة مميزات اكثر عليها ...

الدرس السادس

اخوتي الكرام لقد رأينا سابقاً كيف يمكن ان نكون برنامج محادثة لاكثر من شخص ولكن كان ذلك بمثابة البداية حيث لم يحتوي على اشياء مطلوبة بشده وهي ظهور قائمة بالموجودين حاليا كما انه يجب ان يعرض تنبيهاً اذا غادر اي شخص المحادثة لذى نقوم هنا بتعديل البرنامج السابق كي يتضمن هذة الميزة المهمة ....

سوف نضيف ListBox الى كلاً من الخادم والعميل وذلك بغرض وضع الاسماء داخل هذة القائمة ثم نجعل العميل عند اتصاله بالخادم مباشرة يرسل رسالة فيها الامر NewUser ثم اسم المستخدم الجديد وعندما يستلم الخادم هذة الرسالة يقوم مباشرة باضافة العميل الى قائمة المستخدمين عنده ثم يرسل رسالة الى جميع العملاء الاخرين عنوانها NewUser وبها اسم العميل الجديد ثم يرسل ايضاً رسالة اخرى هي NewList وبها اسماء الاعضاء جميعاً بحيث يقوم كل عميل عندما يستلم هذة الرسالة بمسح القائمة التي لديه ثم يضيف اليها الاسماء التي جاءت في رسالة الخادم .....
الامر الى هنا يبدو سهلاً ولكن كيف يمكن ان ارسل محتويات القائمة ونحن نعلم ان القائمة بها اكثر من اسم ونحن نريد ان نرسل رسالة بها كل اسماء المستخدمين بحيث لا تختلط هذة الاسماء مع بعضها البعض ويتمكن العميل من استجراج جميع الاسماء واضافتهم الى القائمة الخاصة به .... طبعاً نحن لن نقوم بارسال الاسماء في القائمة واحداً بعد الآخر ولكننا سوف نرسل جميع الاسماء دفعة واحدة بحيث نقوم بالفصل بينهم بفاصل معين ولنقل مثلاً الرمز "#" ... وبذلك اذا كان لدينا قائمة بها اربعة مستخدمين هم علي و وليد وهيثم و ابوعابد فان الخادم يقوم بجعل الرسالة كالتالي : "علي#وليد#هيثم#ابو عابد" ويضيف لها الامر NewList بحيث يعلم العميل بانه يجب علية ان يضعهم في قائمة جديدة بعد ان يفصلهم عن بعضهم البعض ... ولكي نقوم بعملية تكوين هذا الشكل من قائمة المستخدمين في الخادم نستخدم كود كالتالي وقد اسميته هنا: GetList

```
Private Function GetList() As String
    Dim S As String
    Dim I As Integer

    S = ""
    For I = 0 To lstUsers.ListCount - 1
        S = S + "#" + lstUsers.List(I)
    Next

    GetList = Mid(S, 2)
End Function
```

حيث قمنا بعمل دوران لكل العناصر في القائمة lstUsers والتي تحتوي على كل المستخدمين في الخادم ووضعنا قبل كل اسم الرمز... "#"
طبعاً عندما تصل هذة الرسالة الى العميل وهي تحمل الامر NewList سوف يكون على العميل ان يقوم بفصل جميع الاسماء عن بعضها البعض ثم يضعها في القائمة عنده ولنفترض ان اسم القائمة في العميل هي lstUsers ايضاً .. سوف نحتاج لفصل الاسماء الى الامر Split وهو يقوم بفصل اي نص يحتوي على فاصل معين وهو في مثالنا "#" ثم يضع الناتج في مصفوفة وبذلك يمكننا ان نضيف عناصر هذة المصفوفة الى القائمة lstUsers كالتالي وقد سميته هنا: FillList

```
Private Sub FillList(S As String)
    Dim A() As String
    Dim I As Integer

    On Error Resume Next
    A = Split(S, "#")

    lstUsers.Clear
    For I = LBound(A) To UBound(A)
        lstUsers.AddItem A(I)
    Next
End Sub
```

حيث ان الاسماء تكون موضوعه ضمن المتغير S ثم نقوم باستخدام الامر Split لفصل الاسماء ووضعها

في مصفوفة من النوع String واسمها A ثم بعد ذلك نقوم باضافة عناصر هذة المصفوفة A الى القائمة lstUsersوبذا نكون قد استلمنا القائمة الجديدة كاملة ....

هذا بالنسبة لارسال واستلام الاسماء ولكن ماذا عندما يقرر احد العملاء ان يغادر فانه يجب ان نقوم بحدفه من القائمة ونحن نعرف انه لكي نحذف عنصر من اي قائمة نستخدم الامر RemoveItem كما سبق ان اوضحنا في السابق وهو يحتاج الى رقم العنصر الذي نريد حذفه ولكننا ليس لدينا سوى اسم العنصر وليس رقمه لدى يجب ان نحصل على رقم العنصر في القائمة بواسطه اسمه الذي معنا وسوف نستخدم لذلك عملية بحث داخل القائمة عن اسم المستخدم ثم عندما نعثر عليه نأخذ رقمه ونحذفه وقد جعلت لذلك امر باسم RemoveUser ويتم اعطائه اسم المستخدم وهو يتكفل بالبحث عنه ثم حذفه من القائمه وذلك كالتالي :

```
Private Sub RemoveUser(User As String)
    Dim I As Integer

    For I = 0 To lstUsers.ListCount - 1
        If lstUsers.List(I) = User Then
           lstUsers.RemoveItem I
        End If
    Next
End Sub
```

طبعاً يستخدم هذا الكود العميل عندما تصله الرسالة DelUser متبوعه باسم المستخدم والتي تعني ان المستخدم فلان قد غادر المحادثه فيقوم باستخدام الاجراء السابق لحذفه من القائمة كما يقوم بعرض تنبيه للعميل بمغادرته للمحادثة .....

من الملاحظ انه في كلا مما سبق ان المسألة تعتمد بشكل كبير على التعامل مع القائمة من حذف واضافة وبحث لدى يجب علينا تفحص الكود السابق جيداً بحيث نستوعبه كاملاً لان التمارين القادمه التي سوف اضعها بعد ان نناقش اي غموض في الموضوع ان شاء الله تعالى ستكون صعبه شوي لذى يجب الاستعداد لها جيداً ...

التعامل مع الملفات

بسم الله الرحمن الرحيم ...

في الواقع لا نستطيع الخوض في ارسال الملفات دون معرفة التعامل مع الملفات بواسطة الغيجوال بيسك حيث انه يجب علينا ان تمكن من فتح وانشاء وحفظ ملف بواسطة الكود ثم بعد ذلك نتطرق الى عملية الارسال التي لن تشكل مشكله بعد ذلك ...

ويجب ان نعلم انه توجد طرق محددة للتعامل مع الملفات في الفيجوال بيسك سوف نستخدم نحن الطريقة Binary وتتميز هذة الطريقة عن الطرق الاخرى في كونها تتعامل مع الملفات كبيانات اولية وليس كنوع معين مثل Input و Output اللتان تتعاملان مع ملفات النصوص او Random والتي تتعامل مع انواع ينشئها المبرمج اما الـ Binary فهي تتعامل مع كل الانواع ولدى فهي ممتازة لنقل اي نوع من الملفات سواء كان ملف صوت او صورة او قاعدة بيانات او غير ذلك وبغض النظر عن نوعه...

فلكي نفتح ملف نستخدم الصيغة التالية :

```
Dim F As Integer

F = FreeFile
Open FileName For Binary As F

Close F
```

لاحظ اننا في السطر الاول عرفنا متغير F من النوع Integer حتى نخزن فيه رقم الملف المفتوح وبذلك يمكن بعد ذلك ان نستخدم رقم الملف بدلاً من اسمه مثلاً اذا اردنا ان نغلق الملف نكتب Close F وتعني اغلق لنا الملف الذي رقمة مخزن في المتغير F والدالة FreeFile تقوم بالبحث عن رقم ملف غير مستعمل اي كاننا نقول ابحث عن رقم جديد وضعه
في المتغير F لكي نستخدمه مع الملف ثم بعد ذلك نستخدم الامر Open لفتح الملف الذي اسمه FileNameمن النوع Binary ونتعامل معه كانه الرقم F وفي الاخير قمنا بإغلاق الملف F بواسطة Close F ....

طبعاً المثال في السابق لم نقوم بعمل شيء سوى اننا فتحنا الملف FileName ثم قمنا باغلاقه بعد ذلك ... (لاحظ انه اذا كان الملف الذي اسمه FileName غير موجود في القرص الصلب يقوم الفيجوال بإنشاء ملف جديد بهذا الاسم)...

ولكي نستطيع ان نكتب الى هذا الملف نستخدم الامر Put ولكي نقرأ منه نستخدم الامر Get مثلاً لكي نكتب الى الملف العبارة "على الله توكلنا" نستخدم التالي :

```
Dim F As Integer
Dim S As String

S = ""على الله توكلنا
F = FreeFile

Open FileName For Binary As F
  Put #F, 1, S
Close F
```

لاحظ اننا وضعنا العبارة في متغير نصي ثم فتحنا الملف ثم استخدمنا الامر put لادخال النص في الملف

الذي رقمه F وبداية الكتابة في الموقع رقم ١ ثم اخيراً نضع محتويات المتغير ... S حيث انه لكي نستخدم الامر Put نحتاج الى رقم الملف ثم الى الموقع الذي نريد الكتابة فيه ثم الشيء الذي نريد كتابته لاحظ انك اذا فتحت الملف مرة اخرى ثم استخدمت عبارة اخرى ووضعتها في الموقع رقم ١ فانه سوف يتم الكتابة فوق العبارة السابقة لذى لكي نضيف العبارة الجديدة الى الملف بعد العبارة السابقة يمكن ان نحدد موقع الكتابة في مكان آخر بعد العبارة السابقة ولكي نعرف طول الملف نستخدم الامر LOFوالذي يرمز لـ Length Of File اي طول الملف ثم نضيف النص الجديد تخيل الان اننا نريد ان نضيف العبارة "ولا حولة ولاقوة الا بالله" نفتح الملف ثم نكتب بعد العبارة في الموقع الذي يلي العبارة السابقة كالتالي :

```
Dim F As Integer
Dim S As String

S = ""ولا حولة ولاقوة الا بالله
F = FreeFile

Open FileName For Binary As F
  Put #F, (LOF(F) + 1), S
Close F
```

لاحظ التغيير الحاصل في الموقع حيث استخدمنا الامر LOF لمعلافة طول الملف F وطوله هو ١٦ لان العبارة السابقة التي يحويها الملف طولها ١٦ حرف ثم بعد ان وجدنا طول الملف قمنا باضافة رقم واحد وبالتالي فان الكتابة سوف تتم في الموقع الذي يلي العبارة السابقة ببايت واحد وبذلك نضمن اضافة العبارة الجديدة الى العبارة السابقة .... وسوف نستخدم نفس المبدء في عملية نقل الملفات اذ اننا سوف نقوم بنقل البيانات على دفعات ونقوم باضافة البيانات الى الملف دفعة بعد الاخرى كما فعلنا في المثال السابق لذى يجب استيعابه جيداً....

الان ما الذي يجب علينا عمله اذا اردنا القراة من الملف مثلاً اذا اردنا قراءة ١٠ حروف من الملف فاننا سوف نقوم بالتالي :

```
Dim F As Integer
Dim S As String

S = Space(10)
F = FreeFile

Open FileName For Binary As F
  Get #F, 1, S
Close F
```

لاحظ الذي عملناه في المتغير S قمنا باستخدام الدالة Space والتي تجعل النص بطول معين من الفراغات ونحن جعلناه هنا بطول 10 حروف فارغة وهو عبارة عن المخزن للبيانات التي سوف نقرأها فاذا اردنا ان نقرأ خمسة حروف فقط نستخدم Space وبين قوسين الرقم ٥ ثم بعد ذلك فتحنا الملف واستخدما العبارة Get للقراءة من الموقع رقم ١ بطول المتغير S والذي هو هنا ١٠ حروف اي ان النص الذي سيكون في المتغير S بعد العلمية السابقة هو "على الله " وذلك الان القراءة من بداية الملف بطول عشرة حروف واذا اردنا ان نقرأ الملف كله نستخدم التالي:

```
Dim F As Integer
Dim S As String

F = FreeFile

Open FileName For Binary As F
   S = Space( LOF(F) )
   Get #F, 1, S
Close F
```

لاحظ اني استخدمت طول المتغير الذي يخزن البيانات هو LOF(F) وهو كما عرفنا فهو طول الملف الذي رقمه F ويمكنك ان تلاحظ اني نقلت العبارة Space من الخارج الى بعد العبارة Open وقبل العبارة Close وذلك الان الامر LOF يعني ماهو طول الملف F ولو جعنا العبار قبل السطر Open فان الملف F سيكون لم يفتح بعد لذى لن يعرف ماهو F وسعطي خطأ لدى وضعت العبارة بعد السطر الذي يفتح الملف .... F لذى فان العملية سوف يتنج عنها ان المتغير S سوف يكون بطول الملف كله وبعد ذلك استخدمنا Get للقراءة من الموقع الاول بطول S يتنج ان محتويات الملف كاملاً سوف يتم وضع نسخة منها في المتغير S وبعدها يمكن ان نعرض محتويات المتغير S في مربع نص مثلاً ... كالتالي :

```
Text1.Text = S
```

اخوتي يجب ان تجربوا عمليات مختلفة على الملفات حتى تتكون لديكم فكرة قوية عن البرمجة الصحيحة فلايجوز ان يجهل احدنا كيف يتعامل مع الملفات بواسطة الكود وبعيداً عن عملية نقل الملفات بواسطة الوينسوك ان التعامل مع الملفات مهم جداً لعمليات التخزين او التشفير او التجزئة او حفظ واسترجاع البيانات فقد تحتاج يوماً ان تبني لبرنامجك نوع معين من الملفات بامتداد خاص بك وتقوم بتخزين بيانات خاصة بك فيه واجرا العمليات المختلفة عليها واظن ان الامر يستحق بعض العناء في تعلمه و لايقل اهمية عن موضوع درسنا في الربط بين الاجهزة لذى قم بالبحث في المنتدى عن اي مواضيع او اكواد قد تفيدك في التعامل مع الملفات بينما اقوم انا بالتحضير للدرس الذي يليه ..

الدرس السابع : التعامل مع الملفات (ارسال واستقبال)

علمنا سابقاً كيف يمكن ان نفتح ملف ونقراء ما فيه بشكل كامل LOF ويمكن ان يتبادر الى الذهن ان الموضوع منتهي وبما اننا نعرف كيف يمكن ان نرسل النصوص عن طريق SendData فنحن نستطيع بذلك ان نقوم بفتح الملف ثم نخزن جميع محتوياته في المتغير S ثم بعد ذلك نستخدم الامر SendData لارسال البيانات وعندما تصل الى الطرف التالي يقوم بفتح ملف جديد ثم يخزن البيانات التي وصلته داخله عن طريق Put بدلاً من عرض الرسائل كما كان في برامج المحادثة السابقة ... اليس كذلك ؟

طبعاً الاجابة هنا نعم ... ولكن الامور لا تجري بهذا الشكل في الظروف الحقيقة حيث ان سرعة الانترنت او الشبكات ليست كما تراه الان عندما تجرب البرامج داخل كمبيوترك لذى فقد تقوم بفتح ملف كبير ثم ترسل بياناته دفعة واحدة وهذا خطأ كبير يرتكبه البعض في ارسال الملفات وان كان يعمل بعض الاحيان فهو لن يعمل في احيان اخرى ... خذ مثلاً نحن جميعاً نستخدم الماسنجر .. فكم مرة ظهرت لك الرسالة تعذر ارسال الملف ؟؟؟ ذلك على الرغم من ان الماسنجر هو برنامج صممته مايكروسوفت ! ... فلماذا هذا العيب الذي يظهر فيه احياناً ؟ طبعاً هناك امور مؤثرة كثيرة واهمها اختلاف سرعة التوصيل ففي بعض الاحيان عندما نقوم بتحميل ملف من الانترنت نجد ان سرعة التحميل هي مثلاً ٥ KB في الثانية وقد تستغرب اذا قلت لك اني في احدى المرات وجدت ان سرعة التحميل لا تتعدى ٧ بايت في الثانية !!! ... ولك ان تتخيل كم من الوقت يحتاج نقل ملف بطول ١ ميجا بايت بهذة السرعة !! مثلاً اذا كانت سؤعة النقل هي ١ كليوبات في الثانية فهذا يعني انك تحتاج الى اكثر من ١٧ ساعة لنقل ملف بطول واحد ميجا بايت ....

مما سبق تظهر الحاجة الى استخدام تقنية اخرى لتلافي هذة المشكلة وهي في هذة الحالة تقسيم الملف الى دفعات ثم بعد ذلك نرسله بشكل متتالي الى الطرف الاخر مع تمكين الطرف الاخر من ان يلغي الاستقبال في اي لحظة يريد ... وهذا يضمن اولاً عدم الضغط على الشبكة ثم ان عملية الارسال تكون سريعة ذلك ان معالجة كمية كبيرة من البيانات تأخذ وقت وجهد كبيرين على الجهاز مما يشكل ضغط يوثر على عمل باقي البرامج وقد لايتم استلام البيانات اصلاً . بينما الارسال المتتالي يسمح للنظام بان ينظم العمليات بشكل سليم مع عدم اعاقة عمله ...

الان اصبح ليدنا اسبابنا كي نرسل الملف بشكل متتالي بدلاً من ارساله دفعة واحدة وبقي ان نعرف كيف نترجم ذلك برمجياً ...

دعنا الان نفترض حجم ملف تخيلي مثلاً ١٨ بايت واردنا ارساله بطول ستة بايت في كل مرة يعني ذلك اننا يمكن ان نستخدم دوران عن طريق For Next او Do While او ما شابه ونجعله يدور ثلاث مرات في كل مرة نقرأ من الملف بيانات بطول ستة بايت عن طريق S=Sapce(6) ثم الامر Get وبعدها نرسل عن طريق SendData S هذا جميل ... ولكن ماذا اذا كان طول الملف هو ٢٠ وليس ١٨ فاننا هنا سوف نعمل شيء قريب من هذا :

```
    Dim S As String
    Dim I As Integer
    Dim F As Integer

    F = FreeFile
    Open FileName For Binary As F

    For I = 1 To LOF(F) \ 6
        S = Space(6)
        Get #F, , S
        wsk.SendData S
        DoEvents
        Print S; "END"
    Next

    If LOF(F) Mod 6 Then
       S = Space(LOF(F) Mod 6)
       Get #F, , S
        wsk.SendData S
        DoEvents
    End If

    Close F
```

اذا دققنا في الكود السابق سوف نجد اننا استخدمنا اشياء غريبة بعض الشيء مثل LOF(F) \ 6 وكذلك LOF(F) Mod 6وكل ذلك بسبب اننا نريد ان نعرف كم حجم البيانات المناسب الذي نريد قراتها من الملف خذ مثلا في مثالنا السابق كانت البيانات بطول ١٨ بايت ونريد ان نقرا بطول ستة لان العدد ١٨ يقبل القسمة بدون باقي على ٦ فأن الاجزاء التي سوف نقرأها ستكون كالتالي ٦ + ٦ + ٦ = 18 وهذا يعني امكانية استخدام S=Sapce(6) لمدة ثلاث مرات وينتهي العمل لكن اذا كان الطول هو ٢٠ فان ذلك يعني ٦ + ٦ + ٦ + ٢ = ٢٠ اي اننا يجب ان نقرأ بطول ستة ثلاث مرات اما المرة الاخيرة فاننا يجب ان نقرأ بطول ٢ وليس ستة يعني S=Space(2) وذلك لاننا اذا قرأنا المرة الاخيرة بطول ستة فان حجم الملف الجديد سوف يكون ٢٤ وليس ٢٠ وهذا يعني ان الملف الجديد لن يكون نسخة طبق الاصل من الملف المرسل وهذا ما لا نريده ولكي نتجنب ذلك يجب ان نعرف كم مرات نقرأ بالطول العادي(سته) ثم ما هو الطول الاخير المتبقي كي نجعل طول المتغير S يتناسب معه ... اذا امعنا النظر في ٦ + + 6 + 6 20 = 2نجد اننا نقرا ثلاث مرات بشكل طبيعي وهذا يمكن ان نعرفة عن طريق قسمة طول الملف ٢٠ على طول الاجزاء التي نريد ارسالها وهي ٦ ولكن القسمة تكون بدون باقي وهذا سوف يعطي ثلاث لان القسمة مع الباقي تعطي ٣,٣٣٣٣٣٣٣ اي ثلاثة واجزاء كسرية ولكي نقسم في الفيجوال بيسك بدون باقي نستخدم (\ ) وليس الاشارة العادية للقسمة (/) خذ هذا المثال :

```
12 / 8 = 1.5    مع الباقي
46 / 10 = 4.6   مع الباقي

12 \ 8 = 1      بدون باقي
46 \ 10 = 4     بدون باقي
```

وبذلك نكون قد عرفنا عدد المرات التي نقرأ فيها بطول عادي وهو سته بايت وانظر الى الكود ستجد اني استخدمت للدوران For من واحد وحتى LOF(F) \ 6 وذلك لان طول الملف يأتي مخزون في LOF كما عرفنا سابقاً وهذا يضمن ان يتغير العدد بتغير طول الملف فاذا كان الملف طوله LOF=15 فأن الناتج من قسمته بدون باقي على سته هو ٢ واذا كان طول الملف ٣٤ فان الناتج سيكون ٥ وبهذا نضمن مرونة الكود بشكل يناسب كل الملفات ...

وبقي ان نحسب الجزء الاخير من الملف ويمكن حسابه عن طريق الامر Mod حيث انه يعود بباقي القسمة اي ان (٢٠ ثم mod ثم ٦(يعطي الناتج ٢ ويمكن بعدها ان نجعل طول المتغير = S

Space(LOF(F) Mod 6)وهو ما عملناه بعد انتهاء الدوران For Next في الكود السابق حيث قمنا بالتالي :

```
If LOF(F) Mod 6 Then
   S = Space(LOF(F) Mod 6)
   Get #F, , S
    wsk.SendData S
    DoEvents
End If
```

حيث قمنا بالتاكد انه يوجد باقي في السطر If وذلك لانه اذا لم يبقى باقي فلاداعي لتاكد مثل اذا كان طول الملف هو ١٨ فان ناتج القسمة هو ٣ والباقي من القسمة هو ٠ ....

ثم بعد ذلك قمنا بحجز طول المتغير S بحسب الجزء المتبقي من الملف ثم قرانا البيانات من الملف وارسلناها باستخدام....SendData

في مثالنا السابق نلاحظ اننا استخدمنا الرقم ٦ بايت ونحن في الحقيقة لن نستخدم هذا الرقم ولكننا سوف نستخدم طول مختلف ممكن نستخدم طول 2 كليوبايت او اربعة كليوبايت وهي ارقام مناسبة ويمكنك تغيرها اذا رايت ان ذلك يتناسب مع سرعة خدمة الشبكة الاعتيادية لديك ونحن سوف نستخدم الطول ٤ كليوبايت في دروسنا القادمة اي ٤٠٩٦ بايت ....

واخيراً نجد اننا لم نجعل كود يمكن ان يوقف العملية في المنتصف اي قبل ان تنتهي حيث ان الدوران يستمر حتى نهاية الملف ولكن هذا غير مقبول اذا كان الملف كبيرا واراد المستخدم الغاء العملية لذى سوف نضيف شرط يتأكد من العملية لا يراد لها ان تنتهي بحيث يتمكن المرسل والمستقبل من انهاء عملية التراسل متى ارادوا ذلك ....

والمثال على هذا الدرس جاهز الان ولكن بقي ان اكتب له شرحاً مناسباً حيث انه معقد بعض الشيء ويعتمد على كثير من الرسائل التي يتبادلها العميل والخادم كي تتم العملية بشكل سليم ....

تكملة ...

بسم الله الرحمن الرحيم ....

المثال الذي نضعه هنا هو مثال اولي لعملية ارسال واستقبال الملفات بحيث نستمر بعد ذلك في الدروس القادمة بتطويره كي نصل الى عملية ارسال واستقبال اكثر موثوقية ...

سوف نبدء الان في مناقشة العمل الذي يقوم به برنامج العميل وهو في درسنا هذا المسؤول عن ارسال الملفات بينما يقوم الخادم بدور المستقبل ...

يوجد في برنامج العميل اربعة ازرار هم : تحديد الملف ، اتصال ، طلب ارسال و إلغاء الارسال ...

في البداية يقوم العميل باختيار الملف الذي يريد ارساله من القرص الصلب عن طريق الزر تحديد الملف فيظهر له مربع الحوار فتح فيختار الملف المراد ارساله ... ويجب طبعاً النقر على الزر اتصال حتى ينشأ الاتصال بين الخادم والعميل ويتأكد من ان الحالة هي "متصل" ...

بعد ذلك ينقر على الزر طلب ارسال وبهذا يتم ارسال رسالة الى الخادم محتوية على الامر NewFile ثم اسم الملف الذي يراد ارساله ... طبعاً اسم الملف سيكون خالياً من المسار بحيث اذا كان الملف في برنامج العميل هو

```
C:\Data\All Programs\Game.exe
```

فاننا لن نقوم بارسال العنوان كاملاً وذلك لاننا لا نظمن ان يكون هذا المسار موجود في برنامج الخادم حيث ان البرنامجان سوف يعملان في الحقيقة في جهازين منفصلين وبالتالي الاسم المرفق في مثالنا السابق سيكون Game.exe فقط وسوف تكون الدالة FileNameOnly هي المسؤلة عن فصل الاسم عن المسار وتجدها في برنامج العميل ... وبذلك تكون الرسالة المرسلة الى الخادم عند النقر على الزر طلب الارسال "NewFileGame.exe" حيث ان السبعة الحروف الاوالى هي للامر والباقي لأسم الملف بحيث يقوم الخادم بالفصل عند الاستلام كالتالي:

```
Cmd="NewFile"
Msg="Game.exe"
```

وذلك عن طريق الاوامر Left و Mid كما تعودنا سابقاً .... والمهم هنا ان العميل بمجرد استقبال هذا الامر NewFileيقوم مباشرة بعرض رسالة الى المستخدم يخبره ان العميل يريد ارسال ملف اليه فهل يقبل استلام الملف ام لا (كما هو الحال في الماسنجر عند تناقل الملفات بين المستخدمين) .. ويكون اما المستخدم اما القبول او الرفض فاذا نقر على الزر موافق يتم ارسال رسالة الى العميل هي OkSend تخبره بان يبدا الارسال او في حالة الرفض يتم ارسال الرسالة NoSend ومن هاتين الرسالتين يكون على العميل معرفة الخطوة التالية حيث انه اذا استقبل الرسالة OkSend يعرض رسالة الى المستخدم بانه تم قبول ارسال الملف ثم يبدء بعدها مباشرة بارسال الملف .... اما اذا استلم الرسالة NoSend فانه يقوم بعرض رسالة تفيد المستخدم ان طلب الارسال تم رفضه من قبل الخادم ....

والان سوف نناقش سوياً ماذا يحدث اذا ثم قبول ارسال الملف ... حيث انه يقوم العميل بعد ان يقبل الخادم استقبال الملف باستدعاء الاجراء SendFile والذي سوف يكون المسؤول عن كود الارسال وهو كالتالي :

```
Private Sub SendFile(FileName As String)
    Dim S As String
    Dim F As Integer
    Dim P As Long

    Const Size = 2048

    If wsk.State <> 7 Then
       MsgBox ""لايـوجد اتـصال
       Exit Sub
    End If
```

```
    Command1.Enabled = False
    F = FreeFile
    Open FileName For Binary As F

      For P = 1 To LOF(F) \ Size
          S = Space(Size)
          Get #F, , S

          If Sending = False Then
             GoTo Bye
          End If

          If wsk.State = 7 Then
             wsk.SendData "NewPart" & S
          Else
             MsgBox ""لقد تم قطع الاتصال
             GoTo Bye
          End If

          DoEvents
      Next

      If (LOF(F) Mod Size) > 0 Then
         S = Space(LOF(F) Mod Size)
         If wsk.State = 7 Then
            Get #F, , S
            wsk.SendData "NewPart" & S
            DoEvents
         Else
            MsgBox ""لقد تم قطع الاتصال
            GoTo Bye
         End If
      End If

      If wsk.State = 7 Then
         wsk.SendData "EndFile"
         DoEvents
      End If

Bye:
    Close F
    Command1.Enabled = True
End Sub
```

وسوف نناقشه معاً بتروي الا وعلى بركة الله ....

لاحظ اولاً اننا حددنا حجم القطع التي سوف نرسالها ووضعناها في الثابت Size حيث اننا وكما سبق ان قلنا ان الارسال لن يتم دفعة واحدة وانما على دفعات ونحن هنا حددنا حجم الدفعات بـ ٢٠٤٨ بايت والذي يعني ٢ كيلوبايت ثم قمنا بعد ذلك بالتاكد من انه يوجد اتصال ام لا فان لم يكن هناك اتصال فلا داعي لارسال الملف ونكتفي بعرض رسالة على المتسخدم بعدم توفر اتصال مع الخادم في هذة اللحظة....

ثم قمنا بعد ذلك بفتح الملف الموجود اسمه في المتغير FileName وبعدها بدأنا في الدوران For Next....والذي سوف يقوم بالارسال على شكل دفعات حيث نقوم اولاً بقراة الدفعة الاوالى من الملف عن طريق Get ونخزن البيانات التي نقرأها في المتغير S ثم بعد ذلك نفحص المتغير Sending فاذا كان Falseنوقف عملية الارسال ثم نذهب الى السطر Bye الموجود في الاسفل وفيه يتم اغلاق الملف والخروج بعدها.. والمتغير Sending هو الذي يحدد ان الخادم يريد الاستمرار في استلام الملف حيث انه في الملفات الكبيرة تستغرق العملية وقتاً طويلاً واذا اراد الخادم ان يلغي العملية يقوم بالضغط على الزر

"إلغاء الاستلام" فيقوم بارسال رسالة "Cancel" الى العميل والذي بدوره عند استلامها يجعل قيمة المتغير Sending=False وبما ان هذا المتغير يتم فحصه دائماً قبل ارسال اي دفعة فانه اذا اصبحت قيمته Falseتتوقف العملية مباشرة ويتم عرض رسالة للعميل بان الخادم اوقف العملية ...

نعود الان لاستكمال الاجراء SendFile فكما قلنا بعد التاكد من ان المتغير Sending لايساوي False نقوم بارسال الدفعة بواسطة SendData S وكما تعودنا دائماً اننا لا نقوم باستخدام الامر SendData قبل ان نتاكد من حالة الاتصال هي V اي متصل نقوم مباشرة بالارسال ولاحظ اننا نرسال البيانات مسبوقة بالامر NewPartوالذي يمكن الخادم من معرفة ان هذة الرسالة تحتوي على جزء آخر من اجزاء الملف فيقوم عند استلامها بفتح الملف ثم اضافة البيانات اليه كما عرفنا في درس التعامل مع الملفات ثم انتظار القسم التالي .... وتستمر هذة العملية (ارسال واستقبال الدفعات) مالم ينقر احد المستخدمين على الزر الغاء والذي يحول قيمة المتغير Sending=False وبما ان هذا المتغير يتم فحصة دائماً لذى فبمجرد عدم ايجابيته (False) فان العملية بالكامل يتم الغائها ....

وبعد ان يتم ارسال آخر جزء من الملف بنجاح يقوم العميل بعدها بارسال الرسالة EndFile والتي تعني ان الملف ثم ارساله بالكامل وعندما يستلم الخادم هذة الرسالة يقوم بعرض مربع حوار يفيد المستخدم بان العملية انتهت بنجاح.....

.....

في هذا المثال تطرقنا الى ارسال الملفات بصورة نموذجية على اساس ان الاتصال يكون في احسن حالاته ولكن ذلك غير وارد في كل الاحيان لانه كما ناقشنا سابقاً توجد اختناقات في الشبكة الحقيقية لذى فان المثال سوف يعمل في جهازك او في شبكة داخلية بسيطة ومع ذلك قد تنشأ مشاكل في الارسال في بعض الحالات لذى سوف نناقش مستقبلاً بعض الاساليب المتبعة للوصول الى نتائج افضل في اغلب الاحيان .... كما ان هذا المثال مجرد من متطلبات التناقل الاساسية كاشريط حالة يوضح سير العملية وماهو المقدار التي تم ارساله حتى الان .... وغيره .... وهذا ما سوف نتطرق اليه لاحقاً بإذن الله تعالى والمهم هنا التاكد من استيعاب العملية والكود الموجود في هذا الدرس بشكل كامل بحيث يصبح لدينا اساس قوي يمكن ان نتعمق بعدها في الموضوع معتمدين على المعلومات المناقشة هنا.....

ملاحظة :
ان الملف الذي سوف يتم ارساله يتم استقباله في برنامج الخادم بنفس المجلد الذي يوجد فيه برنامج الخادم اي انك اذا اردت التاكد من نجاح عملية الارسال ابحث عن الملف الذي ثم نقله في مجلد برنامج الخادم .... وسوف نضيف مستقبلاً امكانية تحديد المكان الذي نريد ان نستقبل الملف فيه الى برنامج الخادم باذن الله تعالى ....

اضافات على برامج ارسال الملفات

اما الان واستمراراً في مشوارنا الذي بدئناه معاً دعونا نقوم بتطوير طريقتنا لارسال الملفات وسوف نناقش الان وبإذن الله تعالى كيفية اضافة شريط حالة (ProgressBar) لعملية الارسال كي يعطي كلاء الطرفين فكرة عن المرحلة التي وصلت اليها العملية .... ولقد استخدمت لهذا الغرض شريط للحالة قمت بتصميمه من قبل تجده على هذا الرابط
ويمكنك بالطبع استخدام شريط الحالة التي تقدمه مايكروسوفت مع الفيجوال بيسك وانا لم استخدم شريط الحالة الذي صممته هنا الا لانه يحتوي على النسبة المئوية التي تبين سير عملية ارسال الملف وهو ما لا تجده في الشريط الافتراضي الذي يأتي مع الفجوال بيسك ...

والمهم في الامر ان شريط الحالة يحتوي على ثلاث خصائص مهمة هي Min و Max و: Value

```
Max     هو عبارة عن آخر درجة في الشريط
Min     وهو عبارة عن اول درجة في الشريط
Value   عبارة عن الدرجة الحالية
```

والفكرة بشكل مبسط هي ان نجعل القيمة Min=0 ونجعل القيمة Max تساوي حجم الملف الكلي اي LOF(F)ونجعل القيمة Value=0 في البداية وبعدها عند ارسال كل جزء من اجزاء الملف نقوم بزيادة القيمة Value بحسب حجم الجزء الذي ثم ارساله من الملف وبما ان الجزء الذي يتم ارساله يتم تخزينه في المتغير Msg فان الزيادة للقيمة Value تكون Len(Msg) اي حجم المتغير..... Msg

وبقي الان نعرف كيف نطبق ذلك في برنامجنا ....

نحن نعرف انه في الدرس السابق عندما يقبل الخادم استلام الملف فان العميل يقوم باستدعا الاجراء المسمى SendFile وهو المسؤول عن عملية ارسال الملف على شكل دفعات وعلينا الان تعديل هذا الاجراء كي يتناسب مع الاضافات التي نريدها وهي شريط الحالة الجديد ... ولذلك ينبغي لنا قبل ان نرسل الجزء الاول الى الخادم ان نرسل رسالة توضح له ماهو طول الملف الذي نريد ارساله حتى يغير الخادم الخاصية Max لشريط الحالة عنده بما يتناسب مع طول الملف الذي سوف يتم ارساله حيث نقوم بارسال رسالة باسم "TheSize" متبوعة ببحجم الملف اي LOF(F) وعندما يستلمها الخادم يعرف طول الملف الذي سيتم ارساله فيقوم بجعل الخاصية Max لشريط الحالة مساوية لطول الملف ....

وبعد ذلك في كل مرة يتم ارسال جزء جديد من الملف يقوم كلاء البرنامجين الخادم والعميل بزيادة قيمة الخاصية value التابعة لشريط الحالة مما يعطي صورة واضحة عن تقدم عملية الارسال.....

طبعاً ان كل ما سبق يعتمد على مدى فهمك للتعامل مع شريط الحالة او الـ ... ProgressBar اما مسألة الارسال فهي بيسطة وكما اعتدنا عليها سابقاً لذى في حالة انك وجدت صعوبة في فهم التعامل مع شريط الحالة راجع الرابط السابق الذي وضعته لك كي ترى كيف يتم استخدامه بشكل مبسط ثم بعد ذلك قم بتفحص الدرس الجديد كي ترى كيف استخدمناه في درسنا هذا ....

وسيكون الدرس التالي بإذن الله تعالى يتعلق بإضافة اشياء بسيطة لبرنامجنا من شأنها ان تحسن طريقة ارساله واستلامه وخاصة للملفات الكبيرة ....

طريقة اخرى لارسال الملفات

السلام عليكم ورحمة الله وبركاته....
إخوتي بعد أن نقاشنا فيما سبق طريقة إرسال الملفات بشكل مبسط لا يخلو من العيوب سنحاول معاً أن نحسن من طريقتنا السابقة بحيث نتمكن من زيادة نسبة نجاح عملية الإرسال.... طبعاً إن ما سوف نضيفه هنا لا يضمن في كل الحالات أن يتم النقل بصورة كاملة وذلك نظراً للعوامل المختلفة المؤثرة عليها لذا هي إحدى محاولاتنا لزيادة الكفاءة...

والآن نحن نعرف تربيع).فرق بين النظرية وبين الواقع حيث إنني إذا سألت أحدكم ما هو مقدار الجاذبية الأرضية فأنه سيجيب مسرعاً (٩,٨ متر لكل ثانية تربيع) .. ولكن هل هذا صحيح فعلاً .. في الواقع هو صحيح من الناحية النظرية ولكن واقعياً إن معدل جاذبية الأرض يتأثر بعدة عوامل أخرى مما يجعله يختلف من منطقة لأخرى فقد يكون أعلى أو أدنى من تلك القيمة...

وهذا بالضبط ما يحدث في عملية النقل فهي تتأثر بسرعة الأجهزة المستخدمة وبحجم الذاكرة في كل منها كما إن لخطوط الشبكة دور كبير ولمقدار الضغط المسلط على الشبكة في الوقت الذي يتم الإرسال فيه... لذى مهمتنا هنا هي محاولة التغلب على هذة الظروف او على الاقل اخذها بالحسبان في عملية الارسال لزيادة كفائتها ونجاحها في اغلب الحالات ...

ولكي نتغلب على مشكلة الذاكرة ويمكن خفض حجم الاجزاء التي سوف نرسلها في كل مرة مثلاً استخدمنا نحن الحجم 4 كليوبايت حيث جعلنا الثابت Size=4096 ويمكننا ان نجعله واحد او اثنين كليوبايت مثلاً Size=2048 اي اثنين كيلوبايت ولك ان تغيرها كما يحلو لك ..

والان ان الفكرة التي اعتمدناها سابقاً في الاجراء المسؤول على عملية الارسال SendFile هي ان يتم الارسال بشكل متتالي وبصورة تابثة حيث انه يقوم باستخدام دوارن For...Next للارسال دون مراعاه نجاح عملية الارسال في كل مرة .... ولذى اذا حدثت مشكلة في ارسال اي جزء فانه لن يأخذ ذلك في الحسبان بل سوف يقوم بارسال الجزء الذي يليه وهو شيء غير صحيح في الواقع العملي ....

لذى سوف نستخدم طريقة اخرى وهي ان نصمم اجراء فرعي يجعل البرنامج ينتظر لمدة معينه من الثواني قبل ان يقوم بارسال الجزء التالي ولنقل ٢٠ ثانية وبذى نعطي فرصة لعملية الارسال كي تأخذ وقت كافي والاجرا يكون بشكل يشبه التالي :

```
Private Sub Wait20Seconds()
    Dim T As Single

    T = Timer
    Do Until (Timer - T >= 20)
       DoEvents
    Loop

End Sub
```

والاجراء كما هو واضح يقوم اولاً بمعرفة الوقت الحالي ووضعه في متغير T ثم يقوم بالدوارن الى ان يكون الفرق بين وقت بداية الارسال T والوقت الحالي هو ٢٠ ثانية ثم يخرج وهو يحقق فترة انتظار لمده عشرين ثانية اي انك اذا وضعت الكود التالي في زر امر مثلاً :

```
Private Sub Command1_Click()
    MsgBox ""قبل الانتظار
    Wait20Seconds
    MsgBox ""بعد الانتظار
End Sub
```

فأن الذي سوف يحصل انه سيتم عرض رسالة فيها العبارة "قبل الانتظار" وذلك بمجرد الضغط على الزر ثم بعد ما يقارب عشرين ثانية سيتم عرض الرسالة اخرى فيها العبارة "بعد الانتظار" ونلاحظ انه على الرغم من السطور الثلاثة هي سطور متتالية ويجب تنفيدها بشكل متتالي وهو ما احصل فعلاً غير انه عندما بدء الكمبيوتر بتنفي اوامر الاجراء Wait20Seconds فانه اضطر للانتظار لمدة ٢٠ ثانية ثم تنفيد السطر الذي يليه وهو الرسالة الثانية ...

ونحن نعرف انه في برنامجنا للارسال نستخدم الامر DoEvenets بعد الامر SendData كي يوفر وقت بسيط للانتظار ولكننا الان بحاجة لوقت اطول لذى سوف نستبدل الامر DoEvents الذي استخدمناه للانتظار بعد ارسال كل جزء من الملف بالامر Wait20Seconds وبذا نتيح فرصة اكبر للبرنامج كي ينفذ عملية الارسال ....

قد يتباذر الى الذهن ان ما سبق هو حل عملي وكافي للتغلب على مشاكل الارسال ولكننا في الواقع قد تغلبنا على جزء من المشكلة ولكننا في الحقيقة اضفنا مشكلة اخرى ويمكنني توضيحها كالتالي لنفرض ان الملف الذي نريد ارساله حجمه هو واحد ميجا بايت يعني ١٠٢٤ كليوبايت فان برنامجنا سوف يرسل الملف على شكل اجزاء بحجم ٢ كيلوبايت اي ان عدد الاجزاء التي سوف يتم ارسالها هي ٥١٢ جزء ولأننا جعلنا الفترة بين كل عملية ارسال لجزء ما والجزء الذي يليه هي ٢٠ ثانية فاننا سوف نحتاج الى مايقارب ثلاث ساعات لكي نرسل هذا الملف !!!!!...

بينما ارسال ملف بهذا الحجم لا يتطلب كل هذا الوقت الخيالي ... ذلك انه ليست عملية ارسال لجزء من اجزاء الملف تتطلب وقت يقدر بعشرين ثانية بل انه ربما تحتاج بعض الاجزاء الى اقل من ثانية واحدة واجزاء اخرى تحتاج ثانية او اثنتين و ..... و .....

حيث ان كل جزء يحتاج الى وقت معين بحسب حالة النظام وحالة الشبكة و باقي العوامل المؤثرة التي ذكرناها سابقاً والتي تحدث في نفس وقت ارسال جزء ما من اجزاء الملف .... ومن هذا ان كل جزء من اجزاء الملف يحتاج الى وقت معين قد يكون اقل بكثير من ٢٠ ثانية او اكثر منها بحسب الظروف المختلفة ...

اذا فأن الذي نحتاج اليه فعلاً هو ان نعرف مقدار الوقت الذي يحتاجه كل جزء ثم نقوم بالانتظار بما يتناسب معه وبذلك لا نهدر الوقت في انتظار عملية ارسال قد تمت فعلاً ....

ولحل هذة المسألة يمكننا الاستفادة من الحدث SendComplete التابع للاداه Winsock وهذا الحدث ينفذ عندما تنتهي عملية ارسال بيانات معينه اي انك عندما تستخدم الامر SendData لارسال بيانات فانه بعد ان يتم تنفيد الارسال يتم استدعاء الحدث SendComplete وهو يفيد بان السطر SendData قد ثم تنفيده بنجاح...

لذى سنقوم بجعل الاجراء Wait20Seconds يتنظر لمدة ٢٠ ثانية بشكل قياسي ولكن اذا حصل ان الحدث SendComplete ثم استدعاه قبل انتهاء العشرين ثانية فان هذا يعني انه لاداعي لمزيد من الانتظار ذلك ان الارسال قد ثم بنجاح ولذى سنقوم بتعديل هذا الاجراء بحيث ينهي الانتظار في حالة انتهاء الارسال بنجاح حتى وان لم تنتهي الفترة المحددة له وهي ٢٠ ثانية ...

ونحتاج بهذا لتعريف متغير عام نسميه مثلاً OK من النوع Boolean والذي يعني True او False ثم نجعل الكود في الحدث SendComplete كالتالي :

```
Private Sub wsk_SendComplete()
    Ok = True
End Sub
```

وهذا يعني ان نجعل قيمة المتغير Ok تساوي True في حالة نجاح الارسال ونقوم بعد ذلك بتعديل الاجراء Wait20Seconds كالتالي :

```
Private Sub Wait20Seconds()
    Dim T As Single

    Ok = False
    T = Timer
    Do Until (Timer - T >= 20) Or Ok
       DoEvents
    Loop

End Sub
```

وبذلك ترى اننا غيرنا قمية المتغير Ok الى False قبل الانتظار ثم قمنا بالانتظار الى ان يتم احد الشرطين وهما ان تنتهي المهلة المحددة وهي ٢٠ ثانية او ان يعمل الحدث SendComplete فيجعل قيمة المتغير Okتساوي True وبذلك ينتهي انتظارنا عندها لان الارسال يكون قد تم فعلاً ولاداعي حينها للتاخير ....

طبعاً هذة الطريقة هي محاولة لتغلب على مشكلتنا في الارسال بالنسبة للملفات الكبيرة وغيرها وهي وان كانت طريقة اكثر قوة من سابقتها فهي لا تعني سلامة الارسال في كل الاحوال ... وقد جربتها انا مع بعض الملفات الكبيرة وعملت بشكل جيد والحمد لله تعالى وسوف نتطرق يعدها الى اضافة بعض المميزات اليها لاحقاً ان شاء الله تعالى ....

ملاحظة بسيطة

نحن كنا في السابق نستخدم الاجراء Wait20Seconds والذي يقوم بالانتظار لمدة عشرون ثانية او حتى يكون المتغير OK يساوي True وهو مايحدث نتيجة الحدث SendComplete بالشكل التالي :

```
Private Sub wsk_SendComplete()
    Ok = True
End Sub
```

وما نحتاجه نحن في هنا ان نجعل المتغير OK تكون قيمته True عند انتها الخادم من حفظ البيانات (او الدفعة الاخيرة) الى الملف وبالتالي فهو يكون على استعداد لاستقبال دفعه اخرى ولذى نقوم بجعل الخادم عند انتهائه من حفظ كل الجزء من الملف بارسال رسالة الى العميل كي يغير قيمة المتغير OK الى True وبالتالي يتم ارسال البيانات الجديدة وجعلت الرسالة بإسم "OKTrue" وبمجرد ان يستلمها العميل يجعل OK=True وهو التغير الوحيد في العميل مما يسبب استكمال ارسال الملف بالتدريج ..

ومن عيوب هذة الطريقة هي البطء النسبي ذلك ان العميل بعد ارسال كل جزء من البيانات يظطر الى الانتظار حتى يستلم من الخادم اشعار باستعداده لاستقبال الجزء التالي وبدى يصبح العمل على شكل تبادل تراسل مستمر بين العميل والخادم ولكنها طريقة تعطي وثوقية اكبر من ناحية سلامة عملية ارسال واستقبال الملفات ...

ونظراً لهذا التبادل المستمر في الرسال افضل في المشاريع الكبيرة ان يتم الاعتماد على اكثر من اداه وينسوك بحيث تكون واحدة خاصة فقط بعملية الاسال والاخرى لتنفيد عمليات تبادل الاومر الاخرى التي يؤديها البرنامج ولا تعتمد على ارسال الملفات ...

وكمثال تجد انك حين تقوم بارسال او استقبال ملف على الماسنجر تبقى لديك القدرة على الاستمرار في الحديث اثناء تقدم العملية طبعاص يمكنك استخدام نفس الاداه (وينسوك) للتشات والارسال في نفس الوقت ولكن ذلك قد يسبب تداخل اوامر الارسال واوامر التشات وهو رأي شخصي مبني على الاعتقاد فقط ....

المهم هنا اننا بعد ماسبق لانريد ان تتكون صورة باننا قد بنينا برنامج ارسال متكامل بل نقول نحن فقط عملنا جهدنا لانجاح العملية بشكل يمكننا من ارسال ملفاتنا في معظم الاحيان وليس جميعها بشكل سليم ان شاء الله تعالى وقد اعتمدت فيما سبق على تجارب شخصية لذى فاني اقول اذا كنا قد عرفنا الكثير عن العمليات السابقة فقد خفي عنا الكثير ايضاً .... واتمنى للجميع حظاً موفقاً مع هذة الاداه .....

ولكني اظن انه بقيت اجزاء مهمة يجب ان نخوض فيها وهي عملية حماية انفسنا من الهكر والذي اعتقد انهم مهما عملوا فهم مبرمجون مثلنا واذا كان لهم عقول سوداء فلنا عقول مسلمة تأبى ان تضر الغير مع عدم عجزها ابداً عن الدفاع عن نفسها بإذن ربها ....

وسوف احاول توضيح بعض الطرق التي قد لاتوفر حماية كاملة ولكنها على الاقل لن تدعنا لقمة سائغة ان شاء الله تعالى ...

واتمنى ان يعيني الله واياكم على رفع شأن ديننا وامتنا الاسلامية عالياً ان شاء الله تعالى ...

الدرس الثامن

السلام عليكم ورحمة الله وبركاته ...

دعونا الان نبين امراً

نحن فيما سبق رأينا كيف يمكن ان ننقل ملفاً باستخدام الوينسوك وقد اعاننا الله على ذلك ... وهي طبعاً ليست الطريقة المثلى ولكنها افضل ما توصلنا اليه سوياً ...

وقد رأيت اهتمام بعض الاخوة بنقل صورة عن سطح المكتب وسوف نناقش هنا الامر بشكل نظري لتوضيح الفكرة ... فلكي نقوم بذلك يجب ان نفكر بسطح المكتب كصورة عادية اذ ان العملية تتم بالطريقة التالية :

١ - ارسال رسالة الى الخادم كي يأخذ صورة لسطح المكتب.
٢ - يقوم الخادم بأخذ الصورة ويحفظها في ملف ..
٣ - يقوم الخادم بأرسال الملف الى العميل ..
٤ - يقوم العميل بتحميل الملف وعرضة في مربع صورة..

التوضيح:

لكي نقوم بأخذ صورة لسطح المكتب يمكن ذلك بعدة طرق ايسرها استخدام الدالة keybd_event وهي دالة API تستخدم بغرض اجبار الكمبيوتر على ارسال رسالة الى النظام بأن مفتاح معين قد ثم الضغط عليه .. واغلبنا يعرف ان الزر Print Screen الموجود في اعلى لوحة المفاتيح يمكن استخدامه لنسخ صورة سطح المكتب ... اليس كذلك ؟؟؟

على العموم ... لمن لايعرف ذلك يقوم بالتالي اولاً قم بالضغط على المفتاح Print Screen الموجود في اعلى يمين لوحة مفاتيح الكمبيوتر ويقوم حينها النظام بأخذ صورة للشاشة متضمنة كل ماتراه انت في تلك اللحظة على الشاشة ويضعها في ذاكرة الكمبيوتر ثم بعد ذلك قم بالانتقال الى برنامج الرسام التابع للوينذوز ومن القائمة تحرير اختار لصق ... سوف تجد صورة سطح المكتب بكل ما يحتويه امامك على شكل صورة يمكنك تحريرها...

ونحن هنا نستخدم نفس الفكرة ففي الاول نقوم بجعل البرنامج يخبر الوينذوز بانه قد قام احدهم بالضغط على الزر PrintScreen وذلك باستخدام الدالة keybd_event وحينها ينسخ الوينذوز صورة سطح المكتب الى الذاكرة (الحافظة او ... (Clipboard

وبعد ذلك نقوم باستخدام الامر SavePicture لكي نحفظ الصورة الموجودة في الحافظة الى القرص الصلب وبهذا يكون لدينا ملف فيه صورة لسطح المكتب وبعدها يمكننا ارساله الى العميل كما نفعل دائماً مع اي ملف عادي ... وعندما ينتهي ارسال الملف يقوم العميل بتحميل الملف الى مربع صورة كي يراه المستخدم وذلك عن طريق الامر LoadPicture الذي يستخدم لتحميل صورة من ملف الى مربع الصور ...

سوف تجدون في الاسفل برنامج بسيط يقوم بعمل وضع صورة سطح المكتب في ملف صورة ومن ثم يقوم بعرض هذا الملف ... وهو كالتالي :

```
Private Declare Sub keybd_event Lib "user32" (ByVal
bVk As Byte, _
ByVal bScan As Byte, ByVal dwFlags As Long, ByVal
dwExtraInfo As Long)
Private Const VK_SNAPSHOT = &H2C

Private Sub Command1_Click()
    keybd_event VK_SNAPSHOT, 0, 0, 0
    DoEvents
```

```
    SavePicture Clipboard.GetData(), App.Path &
"\Desktop.bmp"

    Set RgheedViewBox1.Picture = LoadPicture(App.Path
& "\Desktop.bmp")
End Sub
```

في البداية تعريف الدالة keybd_event وكذلك الثابت VK_SNAPSHOT وهو عبارة عن رقم الزر PrintScreenحيث ان لكل زر رقم معين ...

وفي السطر الاول في الشفرة التي في زر الامر نقوم بأخذ صورة لسطح المكتب ثم نستخدم الامر DoEventsلكي ننتظر حتى انتهاء العملية ...

بعدها نستخدم الامر SavePicture لحفظ الصورة التي في الحافظة ClipBoard.GetData الى داخل الملف.... Desktop.bmp

بعد ان انتهينا من حفظ الصورة في الملف نقوم بعرض الصورة في مربع الصور.... RgheedViewBox1

لاحظ ان مربع الصور التي استخدمته لعرض الصورة هو ليس مربع الصور الاعتيادي PictureBox وانما هو عبارة عن اداة بسيطة لعرض الصور وهي تشبه مربع الصور العادي مع بعض الاضافات حيث ان الاداة تمتلك ميزة ظهور اشرطة تمرير اذا كانت الصورة كبيرة وبالطبع فالصورة كبيرة لهذا استخدمت انا هذة الاداه ويمكنك استخدام الـ PictureBox العادي طبعاً ....

واذا اردت الحصول عل مميزات هذة الاداه الجديدة يمكنك الاطلاع على هذا الرابط :

http://www.vb4arab.com/vb/showthread.php?postid=1000032165&t=6859

طبعاً ان ما سبق كان للتوضيح فقط وفي حالة كونك تريد ارسال الملف باستخدام الوينسوك يفضل دائماً ان يكون حجمه صغيراً لهذا من الافضل كثيراً ان تحول الصورة التي حصلنا عليها لسطح المكتب من التنسيق bmp الى jpg حيث ان حجم الملف مثلاً قبل التحويل هو ١,٣٧ ميجابايت اي حوالي (١٤٠٣ كيلوبايت) بينما بعد تحويلها الى تنسيق jpg فانه يصبح تقريباً ٤٨ كيلوبايت .... وهو حجم صغير جداً مقارنة مع سابقة وبذلك يصبح الارسال اسرع بكثير ...

وسوف اتطرق لاحقاً الى كيفية تحويل الصورة من bmp الى jpg والعكس ...

ان شاء الله تعالى ....

اتمنى للجميع حظاً موفقاً وفي رعاية الله ...